THE MICROCOMPUTER MAGAZINE

commodore

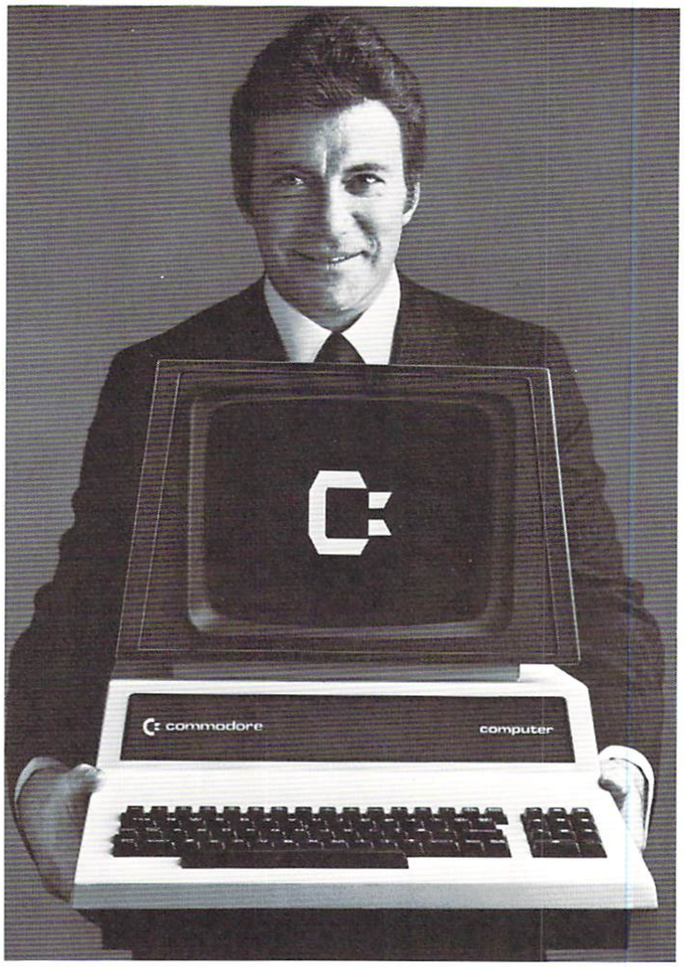
DECEMBER 1981
$2.50

THE COMMODORE COMPUTERS

"FROM $300 TO $1995, THEY COST LESS AND GIVE YOU MORE FOR YOUR MONEY. READ OUR CHART."

—William Shatner

The idea of a computer in every office and home used to be science fiction. Now it's becoming a reality. The question is, with so many to choose from, which computer should you buy? When you consider the facts, the clear choice is Commodore.



COMPARE OUR $995 COMPUTER

FEATURES	COMMODORE 4016	APPLE II	IBM
Base Price	$995	$1,330	$1,565
12″ Green Screen	Standard	299	345
IEEE Interface	Standard	300	NO
TOTAL	$995	$1,929	$1,910
Upper & Lower Case Letters	Standard	NO	Standard
Separate Numeric Key Pad	Standard	NO	Standard
Intelligent Peripherals	Standard	NO	NO
Real Time Clock	Standard	NO	NO
Maximum 5½″ Disk Capacity per Drive	500K	143K	160K

Prices are as of the most recent published price lists, September, 1981 and approximate the capabilities of the (16K) PET® 4016. Disk Drives and Printers are not included in prices. Models shown vary in their degree of expandability.

Many experts rate Commodore Computers as the best desk-top computers in their class. They provide more storage power—up to 1,000,000 characters on 5¼″ dual disks—than any systems in their price range. Most come with a built-in green display screen. With comparable systems, the screen is an added expense. Our systems are more affordable. One reason: we make our own microprocessors. Many competitors use ours. And the compatibility of peripherals and basic programs lets you easily expand your system as your requirements grow. Which helps explain why Commodore is already the No. 1 desk-top computer in Europe with more than a quarter of a million computers sold worldwide.

WE WROTE THE BOOK ON SOFTWARE. The Commodore Software Encyclopedia is a comprehensive directory of over 500 programs for business, education, recreation and personal use. Pick up a copy at your local Commodore dealer.

FULL SERVICE, FULL SUPPORT.
Commodore dealers throughout the country offer you prompt local service. In addition, our new national service contract with TRW provides nationwide support. Visit your Commodore dealer today for a hands-on demonstration.

Commodore Computer Systems 681 Moore Road King of Prussia, PA 19406	Canadian Residents: Commodore Computer Systems 3370 Pharmacy Avenue Agincourt, Ontario, Canada, M1W 2K4

Please send me more information.

Name ____________________

Company ____________ Title ____________

Address ____________________

City ____________ State ________ Zip ________

Telephone ____________________

Interest Area ____________________

☐ Business ☐ Education ☐ Personal

commodore Tractor Printer 4022
commodore computer
commodore dual drive floppy disk

commodore
COMPUTER

THE MICROCOMPUTER MAGAZINE

commodore

FEATURES

DEPARTMENTS

commodore

Editorial Manager
David A. Kaminer
Creative Supervisor
Edythe Silverglide
Editor
Paul Fleming
Staff Editor
Jody Miller
Staff Writers
Paul Goheen
Neil Harris
Mike Heck
Pat McCallister
Jody Miller
Dave Middleton
John O'Brien
Dave Scott
John Stockman
Michael Tomczyk
Contributing Writers
Jim Butterfield
Elizabeth Deal
Sieg Deleu
Paul Higginbottom
Karl Hildon
Preston Marshall
Ed Steinfeld
Technical Staff
Rick Cotton
Paul Goheen
Neil Harris
Pat McCallister
Dave Middleton
Dave Scott
John Stockman
Michael Tomczyk
Circulation Manager
John O'Brien
Advertising Manager
Paul Fleming

Commodore Magazine is issued bi-monthly. Published by the Computer Systems Division, Commodore Business Machines, Inc., 681 Moore Rd., King of Prussia, PA 19406 (215) 337-7100. The U.S. Subscriber rate: $15.00 per year. In Canada and Mexico the Subscriber rate is $25.00. No material may be reprinted without permission. Volume 2, Number 5.

Q&A HOTLINE

Q. I have a 2001-32N Pet upgraded to BASIC 4.0. Will the DOS support program mentioned in my manual and in your newletter Vol.1, issue 9, page 5 be of any use?

D Hankerson, Mankato, MN

A. *If you have DOS 2.X(2.1,2.5,2.6), you can use the disk related ocmmands (dload, dsave, etc.) of BASIC 4.0, and have no need to be continually loading support programs into the top of RAM. The "DOS support" program, also known as "universal wedge" should be found on the test/demo disk in each Commodore disk drive.*

Q. I have a CBM 8032 and an 8050 disk drive and I know enough BASIC to write most of what I need for software. I am just starting a project which will need the ability of creating a data base on the disk (it will create multiple related files). But in doing so, I would like to know if the proposed data base will fit on the disk. I can calculate the number of proposed sectors and files to be used, but I do not know how to access the BAM on the disk to check for free sectors available, nor how to access the directory to obtain the number of free files available. This is important since there are limits to both, and I do not wish to commit the program to the actual initialization of the data base only to have the system 'crash' at some uncompleted stage for having gone out of bounds. How can I access or calculate this information (free sectors and directory file space)?

J. Bowles, Woodland, CA

A. *I can understand your concern and forethought, but writing a program which would tell you when it was running out of room wouldn't really help you get more on a diskette, however there is an efficient and effective way to avoid having the system crash—by using the relative record format. This solution makes your programming job much easier but it does require you to estimate the maximum amount of records to ever exist in the file. If you do completely fill up the file, DOS will tell you and it is a simple matter to copy a relative file into a larger relative file.*

Considering that there are 225 directory entries available on the 8050, it would seem unlikely that you would create too many files in a data base application which generally entails large files.

If you are still insistent on monitoring the number of blocks free on a diskette, such utilities were published in the October 1981 Commodore Magazine on pages 46 and 47.

Q. I've tried unsuccessfully for several weeks to locate copies of the INVADERS, COSMIC JAILBREAK, and COSMIADS programs for my CBM 8032. I know these games are available in England although I don't know if the 8032 versions exist. I would really appreciate it if you could help me.

J. Quass, Cedar Rapids, IA

A. *As far as we know, there are no 80 column versions of these games, but considering that they all run in BASIC 4.0, and do not utilize any part of the 2nd cassette buffer (which cannot be used for program space in the 80 column or new 40 column 12" screen PETs) it would require a minimum of re-assembly to make them work reasonably well.*

Q. I am presently working on a software application for a computerized motel register system using the CBM 8032, 4022 printer, and 4040 disk. I have just purchased the 8032 and I'm having problems finding a full memory map conversion for the 3.0 rom to the 4.0. To be specific, I need to write a machine language that will disable the stop key without disabling the clock. Can you help?

D. Whitmire, Sandston, VA

A. *Here is just such a routine, courtesy of Mike Schaff of Commodore's Mid-West regional office.*

Continued on page 4.

IF YOU'RE WAITING FOR THE PRICE OF WORD PROCESSORS TO FALL WITHIN REASON,

Everyone expected it would happen sooner or later...with WordPro PLUS™ it already has! Now all the marvelous benefits of expensive and advanced word processing systems are available on Commodore computers, America's largest selling computer line. WordPro PLUS, when combined with the new 80 column CBM 8032, creates a word processing system comparable to virtually any other top quality word processor available—but at savings of thousands of dollars!

New, low cost computer technology is now available at a fraction of what you would expect to pay. This technology allowed Commodore to introduce the new and revolutionary CBM 8032 Computer.

WordPro PLUS turns this new CBM 8032 Computer into a sophisticated, time saving word processing tool. With WordPro PLUS, documents are displayed on the computer's screen. Editing and last minute revisions are simple and easy. No more lengthy re-typing sessions. Letters and documents are easily re-called from memory storage for editing or printing with final drafts printed perfectly at over five hundred words per minute!

Our nationwide team of professional dealers will show you how your office will benefit by using WordPro PLUS. At a price far less than you realize.

Invest in your office's future...
Invest in WordPro PLUS...
Call us today for the name of the WordPro PLUS dealer nearest you.

Professional Software Inc.
166 Crescent Road
Needham, MA 02194
(617) 444-5224
TELEX: 95 1579

™WordPro is a registered trademark of Professional Software, Inc. WordPro was designed and written by Steve Punter.

Q&A HOTLINE

```
0000          ;******************************************************
0000          ;*                                                    *
0000          ;*  ssss  ttttttt  oooo  ppppp    k   k eeeee y     y *
0000          ;* s    s    t    o    o p    p   k  k  e      y   y  *
0000          ;* s         t    o    o p    p   k k   e       y y   *
0000          ;*  ssss     t    o    o ppppp    kk    eeeee    y     *
0000          ;*      s    t    o    o p        k k   e        y     *
0000          ;* s    s    t    o    o p    ..  k  k  e        y     *
0000          ;*  ssss     t     oooo  p    ..  k   k eeeee    y     *
0000          ;*                                                    *
0000          ;*                     44     000000                  *
0000          ;*                    4 4     0    0                  *
0000          ;*                   4  4     0    0                  *
0000          ;*                  4   4     0    0                  *
0000          ;*                 4444444    0    0                  *
0000          ;*                      4  .. 0    0                  *
0000          ;*                      4  .. 000000                  *
0000          ;*                                                    *
0000          ;******************************************************

0000          ;******************************************************
0000          ;*
0000          ;* this machine code program is designed to disable the
0000          ;* stop key and all basic commands excluding  run, cont
0000          ;* and goto while continuing to update the clock.
0000          ;*
0000          ;* the attempted use of any other basic  command  other
0000          ;* than the commands  listed  above  will  result  in  a
0000          ;* system warm start.
0000          ;*
0000          ;* to start this program you  must  inclose  the  'sys'
0000          ;* command inside your basic program.  example:
0000          ;*
0000          ;* 10 sys634:rem disable stop key
0000          ;* 20 ......
0000          ;* 30 ......
0000          ;* .
0000          ;* .
0000          ;* .
0000          ;*
0000          ;* to disable this program and restore the use of basic
0000          ;* cammands and the stop key, you must also enclose the
0000          ;* 'sys' command inside your basic program.  example:
0000          ;*
0000          ;* .
0000          ;* .
0000          ;* .
0000          ;* 100 ......
0000          ;* 110 ......
0000          ;* 120 sys670:end:rem end of your basic program
0000          ;*
0000          ;* copyright 1981 by michael schaff
0000          ;******************************************************

0000          ;******************************************************
0000          ;*
0000          ;* programmers note:
0000          ;*      since this program  alters  the  'irq'  of  the
0000          ;*      system, it becomes necessary  to  disable  this
0000          ;*      program before 'chaining' programs together.
0000          ;*
0000          ;*
0000          ;*        author: michael schaff
0000          ;* date written: 09/25/81
0000          ;* listing date: 09/26/81
0000          ;*
0000          ;* copyright 1981 by michael schaff
0000          ;******************************************************
```

```
0000                 clock  =$ffea        ;update the clock
0000                 ivadd  =$e455        ;original interrupt handler
0000                 power  =$fd16        ;jump address basic power up
0000                 curlin =$37          ;current basic line number
0000                 hiv    =$90          ;hardware interrupt vector
0000                 pia    =$9b          ;pia keyswitch


0000                        *=$027a       ;start assemble here


027a                 start
027a  78                    sei           ;set interrupt disable status
027b  a9 85                 lda #<stplop  ;set lo byte of interrupt
027d  85 90                 sta hiv
027f  a9 02                 lda #>stplop  ;set hi byte of interrupt
0281  85 91                 sta hiv+1
0283  58                    cli           ;clear interrupt status to norm
0284  60                    rts           ;return to basic

0285                 stplop
0285  8d a9 02              sta rega      ;store 'a' reg
0288  a5 37                 lda curlin    ;get current line number
028a  c9 ff                 cmp #$ff      ;check if immediate mode
028c  f0 0d                 beq kill      ;yes -- kill program
028e  20 ea ff              jsr clock     ;update clock
0291  a9 ff                 lda #$ff      ;set pia switch
0293  85 9b                 sta pia
0295  ad a9 02              lda rega      ;reset 'a' reg
0298  4c 58 e4              jmp ivadd+3   ;jump to scan keyboard

029b                 kill
029b  4c 16 fd              jmp power     ;kill the program

029e                 basicp
029e  78                    sei           ;set interrupt disable status
029f  a9 55                 lda #<ivadd   ;set low byte of interrupt
02a1  85 90                 sta hiv
02a3  a9 e4                 lda #>ivadd   ;set high byte of interrupt
02a5  85 91                 sta hiv+1
02a7  58                    cli           ;clear interrupt status
02a8  60                    rts           ;return to basic

02a9  00             rega   .byt 00

02aa                        .end


errors = 00000

symbol table

symbol value
 basicp   029e    clock   ffea    curlin   0037    hiv      0090
 ivadd    e455    kill    029b    pia      009b    power    fd16
 rega     02a9    start   027a    stplop   0285

end of assembly
```

Message from the President

Almost a year has passed since I joined Commodore as president. Taking the reigns of one of the world's technological leaders was a task that promised to be both exciting and rewarding; I certainly have not been disappointed in my estimation of the job.

Throughout 1981, Commodore's record breaking momentum was lead by the Computer Systems Division, which accounted for more than 70 percent of sales. As a result, we have continued in our leadership role as a microcomputer manufacturer committed to vertically integrated products.

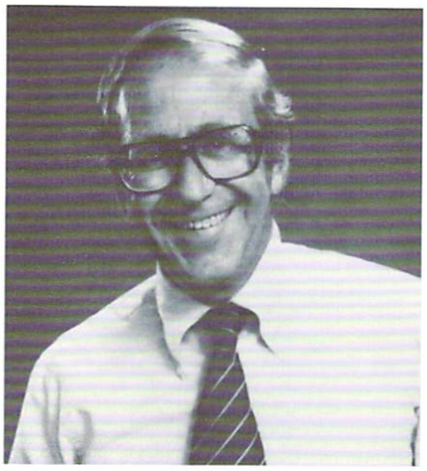
With FCC approval of the VIC 20 in the Spring of 1981, and with increased manufacturing capacity for this low-cost personal computer, Commodore is positioned to capture a significant portion of the entry level personal computer market.

Commodore's participation in the business market continued to grow steadily with the success of the CBM 8000 series business computers. Strong sales of the PET, the world's first self-contained personal computer system, also contributed to Commodore's outstanding performance.

Our SuperPET, combining the power and features of a mainframe computer within the low-cost advantages of a microcomputer, was another exceptional new product introduced during 1981.

Changes in organizational structures are an essential part of responding to the needs of the marketplace. So in 1981, Commodore completed the realignment of its distributor system. A network of eight full-service Commodore owned and operated regions now provide close support to our dealers and customers. More than 600 dealers carry the Commodore line. Additional efforts planned for 1982 should substantially increase this number.

Service is another consideration in purchasing a computer system. A major breakthrough in 1981 was an agree-



ment reached with TRW's Customer Service Division to service and maintain selected Commodore computer products throughout the United States.

In software, we have gained increased recognition with a much greater perception of our product strength. Commodore has gained acceptance in the business sector with our cost-effective word processing capability as well as financial applications, general data handling programs, and specific legal and medical accounting packages.

Commodore's growth as a leading manufacturer of semiconductors and microcomputers is evidenced by our acquisition of a 589,000-square foot facility in the Valley Forge area. This building will accommodate both the manufacturing and administrative functions of the company.

I would personally like to thank all of you who have supported our efforts to expand and improve our strong base of satisfied customers. I hope we have served some of your needs in 1981, and I can promise an even greater commitment both to our dealers and our customers in 1982.■

Jim Finke

James Finke
President

Editor's Notes

This issue is an important milestone in the publication of our young magazine. With this December issue, Commodore Magazine is being published on schedule.

Reader response to the October issue was truly uplifting, generating more positive comments than ever before. But we all know the pitfalls that arise from resting on our laurels. We here at Commodore are committed to publishing a credible user-oriented magazine every two months, without exception. This December issue is one way of saying "we mean business."

In the spirit of the holiday season, I'd like to express my gratitude, on behalf of our entire staff, for the support we received as we worked to establish our existence.

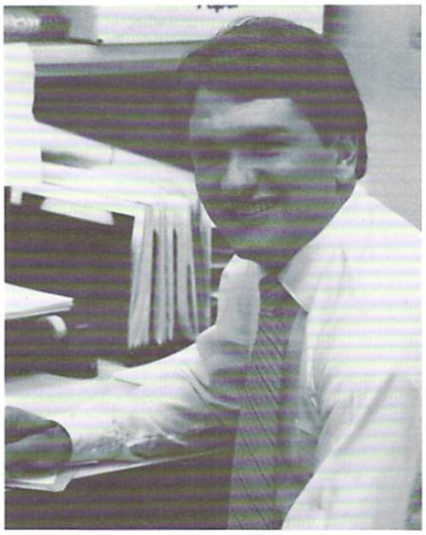
And in anticipation of an exciting new year, we welcome your support in maintaining our hard-earned credibility. We do want your ideas, positive or negative. If you have a technical question, send it to us. We will do our utmost to address your problem in the Q & A Hotline section. If you'd like to share a business or education application story, bring it to our attention—and we in turn will bring it to our readers. Are you a VIC enthusiast? Let us know what you think of our VIC 20 section. And, without a doubt, we can never get enough programmer's tips!

Beginning with this issue we will also review a hardware or software product compatible with our Commodore computers. So if you have a product you feel deserves our attention, just drop us a line.

Obviously, there will be new features and articles that both we, and you, haven't even imagined. But, that's what makes working on this magazine so rewarding, as we try to breathe fresh life into these pages every two months.

As we close 1981 with a positive step forward both for this magazine and Commodore, I can assure you our primary goal in 1982 is to improve with each volume of our new publication.

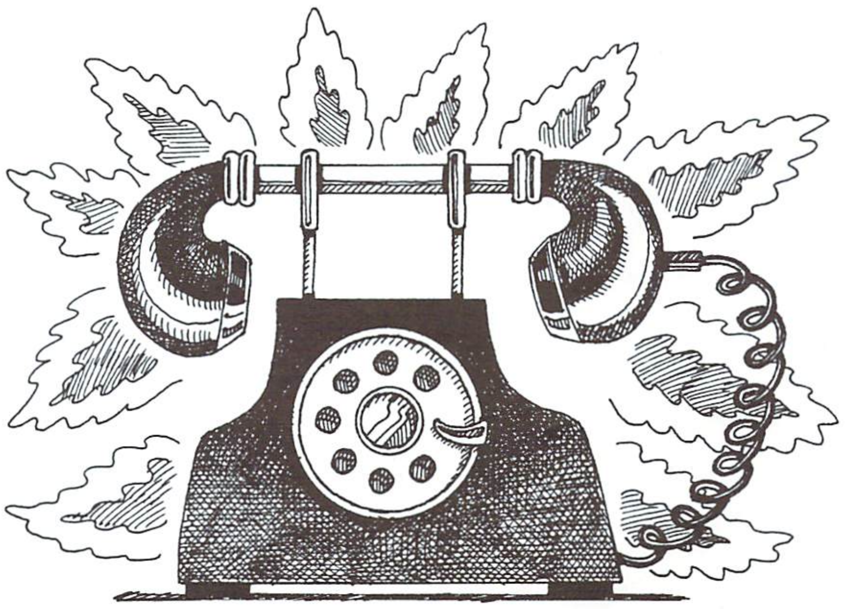
Happy Holidays! ■



Paul Fleming

Paul Fleming
Editor

HOT STUFF!



Hello from the Commodore Hotline! Allow us to introduce ourselves. We're the folks who get a call whenever there's a problem, whether it be software, hardware, magazine subscriptions, documentation . . . the list is truly endless. Our resources for solving your difficulties include a comprehensive library of technical documents, the very informative Software Encyclopedia, our own in-house technical support personnel, and—often most important—a journal of inquiries and responses to previous callers.

It is this last source of information that we would like to share with you in the coming months. We have gained some interestng insights into many of our customers' problems. And, beginning with the February issue of Commodore Magazine, we will tell you about some of the most commonly asked questions—and of course we'll provide the answers.

For this issue, however, we'd like to provide a little background on our activities. The Commodore Hotline (1-800-523-5622) is open for calls from 8:30 a.m. to 5:30 p.m., EST, Monday through Friday. An average of 2000 calls are answered per month, so a lot of people are trying to get through to us. And although we enjoy talking with you, we feel we must limit the amount of time we spend per inquiry. So here are a few suggestions to help you find the source of a problem before you contact the Hotline:

- Refer to your documentation for specific solutions to a problem (you might even try Commodore Magazine, which is quickly becoming a handy reference).
- Check with your local dealer for an answer. Dealers are usually a wealth of information because of their contact with customer problems and due to their access to Commodore's Dealer Alert network. This system provides product news, availability dates, and technical data to dealers on a regular basis.
- Then, if you still need some answers, please feel free to contact us. We'll do all we can to help. ■

USERS CLUBS: Sound Off!

Last issue we told you we were compiling a list of all Commodore Users clubs throughout the country. To date, our list includes the names mentioned below. If you'd like to add your name to the rolls, please send your club's name, address, and other pertinent information to:

Commodore Users Clubs
c/o Editor
Commodore Magazine
681 Moore Road
King of Prussia, PA 19406

And remember, once our list is comprehensive enough, we will begin forwarding valuable information to clubs on a regular basis, including hardware and software updates, technical bulletins, new product announcements, and troubleshooting tips.

CALIFORNIA

Lawrence Hall of Science
UC Berkeley
Computer Project, Room 254
Berkely, CA 94720
(415) 642-3598

Downey-Bellflower Users Group
c/o Robert Johnson
14944 Bayou Avenue
Bellfower, CA 90706

Valley Computer Club
2006 Magnolia Blvd.
Burbank, CA
(213) 849-4094
1st Wed. 6pm

Valley Computer Club
1913 Booth Road
Ceres, CA 95307

PUG of Silicon Valley
22355 Rancho Ventura Road
Cupertino, CA 95014

BAMBUG
1450 53rd Street
Emeryville, CA
(415) 523-7396

North Orange County Computer Club
3030 Topaz, Apt A
Fullerton, CA 92361
Dave Smith

Lincoln Computer Club
750 E. Yosemite
Manteca, CA 95336
John Fung, Advisor

PUG
310 Showers Drive
Mountain View, CA
1st Wed. 7pm
Ford Aerospace Cafeteria
Harry Saal

SPHINX
314 10th Avenue
Oakland, CA
(415) 451-6364
Every 2nd & 4th Thurs.

Midpeninsula PUG
Ford Aerospace Cafeteria
3939 Fabian Way
Palo Alto, CA
(415) 328-7745

Sacramento PET Workshop
PO Box 28314
Sacramento, CA
(916) 445-7926
Every 3rd Thurs-7:30 pm

San Diego PUG
c/o D. Costarakis
3562 Union Street
(714) 235-7626 7 am-4 pm

Walnut Creek PET Users Club
1815 Ygnacio Valley Road
Walnut Creek, CA 94596

Connecticut

Paul W. Sparks
13 Lincoln Drive
Gales Ferry, CT 06335

FLORIDA

Jacksonville Area PET Society
401 Monument Road, #177
Jacksonville, FL 32211

Richard Prestien
6278 SW 14th Street
Miami, FL 33144

South Florida PET Users Group
Dave Young
7170 S.W. 11th
West Hollywood, FL 33023
(305) 987-6982

ILLINOIS

Shelly Wernikoff
2731 N. Milwaukee Avenue
Chicago, IL 60647

Central Illinois PET Owners
Rick Goldsmith
2730 Townway Road #E-54
Danville, IL 61832

INDIANA

PET Users
Jerry Brinson
PO Box 36014
Indianapolis, IN 46236
(317) 898-3604

IOWA

PET Users Group
c/o Don Vorhies
1321 42 St. SE.
Cedar Rapids, IA 52403

MARYLAND

Assoc. of Personal Computer Users
5014 Rodman Road
Bethesda, MD 20016

MICHIGAN

David Liem
14361 Warwick Street
Detroit, MI 48223

PET User Group
Peter Oakes
2235 Lakeshore Drive
Muskegon, MI 49441

MINNESOTA

Twin Cities
John Fung
Twin Cities, MN
(612) 376-5465

MISSOURI

St. Louis Club
Mary Perkinson
46 Westwood Court
St. Louis, MO 63131
(314) 432-5225

NEVADA

Las Vegas PET Users
4884 Iron Avenue
Las Vegas, NV 89110

NEW JERSEY

Amateur Computer Group of New Jersey
John Loofbourrow
UCTI, 1776 Raritan Road
Scotch Plains, NJ 07076
(201) 233-7068

Amateur Computer Group of New Jersey
18 Alpine Drive
Wayne, NJ 07470

NEW HAMPSHIRE

Northern New England
Computer Sociey
PO Box 69
Berlin, NH 03570

NEW YORK

Capital District PET Users
Ben Green
Albany area, NY
(518) 370-1820

Long Island PET Society
Ralph Bressler
Harborfields HS
Taylor Avenue
Greenlawn, NY 11740

PET User Group
Westchester, NY
(914) 428-7872
Every 2nd Tuesday

PET User Group
c/o Meyer
35 Barker Avenue
White Plains, NY 10610

OREGON

NW PET Users Group
John F. Jones
2134 N.E. 45th Avenue
Portland, OR 97213

PENNSYLVANIA

PET User Group
Gene Beals
PO Box 371
Montgomeryville, PA 18936

PACS PET USER GROUP
20th & Olney Streets
Philadelphia, PA

Glen Schwartz
807 Avon
Philadelphia, PA 19116

Gene Planchak
4820 Anne Lane
Sharpsville, PA 15150
(412) 962-9682

TENNESSEE

River City Computer Hobbyists
Memphis, TN
1st Mon. at Main Library

TEXAS

SCOPE
1020 Summit Circle
Carrollton, TX 75006

PET Users
2001Bryan Tower
Suite 3800
Dallas, TX 75201

Larry Williams
PO Box 652
San Antonio, TX 78293

PET User Group
John Bowen
Texas A & M Microcomputer Club
Texas A & M, TX

UTAH

Utah PUG
Jack Fleck
2236 Washington Blvd.
Ogden, UT 84401

VIRGINIA

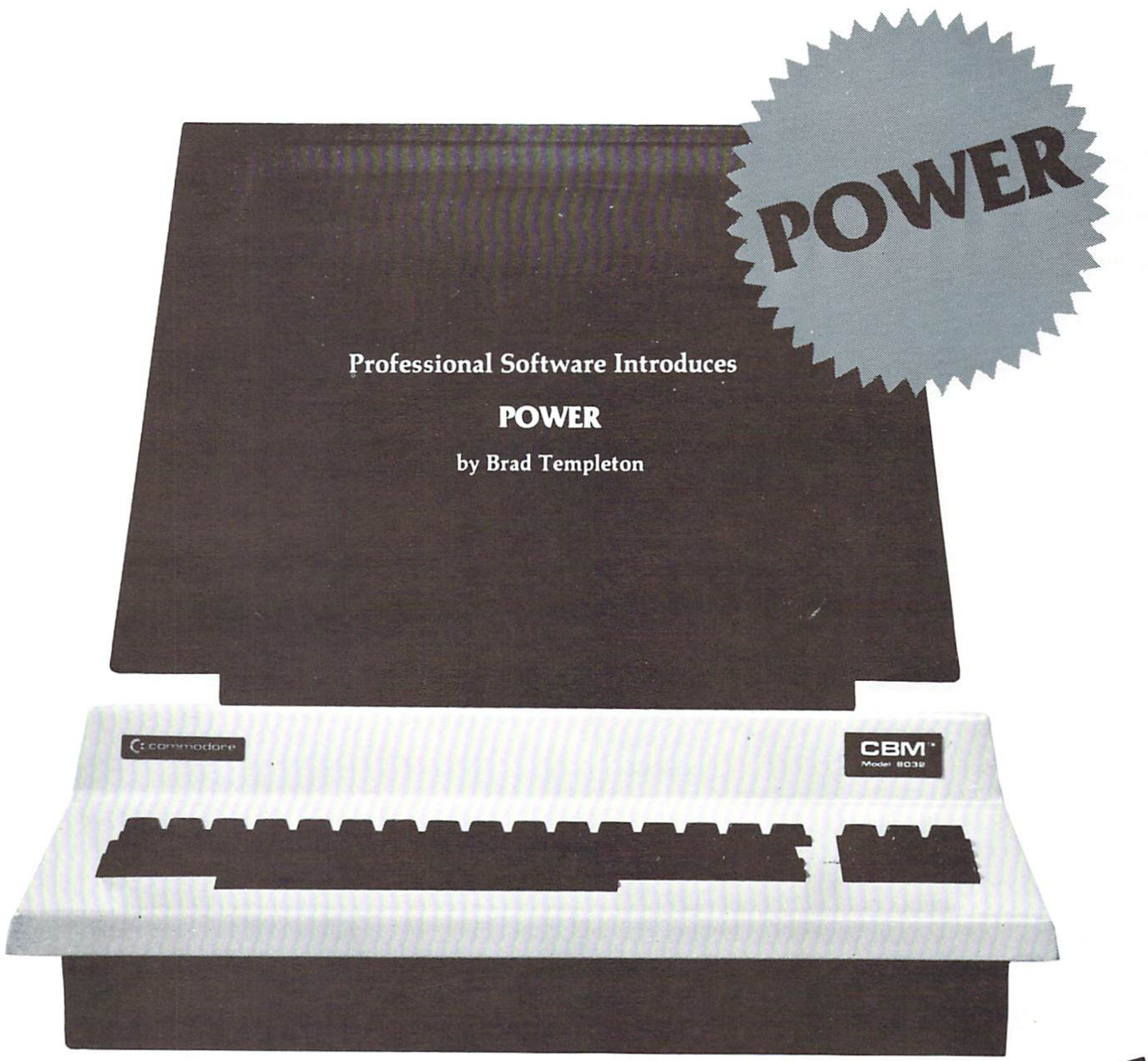
Northern VA PET Users
Bob Karpen
2045 Eakins Court
Reston, VA 22091
(703) 860-9116

WASHINGTON

Northwest PET User Group
PO Box 482
Vashon, WA 98070

WISCONSIN

Sewpus
c/o Theodore J. Polozynski
PO Box 21851
Milwaukee, WI 53221



ADD POWER TO YOUR COMMODORE COMPUTER

$89.95

POWER produces a dramatic improvement in the ease of editing BASIC on Commodore's computers. POWER is a programmer's utility package (in a 4K ROM) that contains a series of new commands and utilities which are added to the Screen Editor and the BASIC Interpreter. Designed for the CBM BASIC user, POWER contains special editing, programming, and software debugging tools not found in any other microcomputer BASIC. POWER is easy to use and is sold complete with a full operator's manual written by Jim Butterfield.

POWER's special keyboard 'instant action' features and additional commands make up for, and go beyond the limitations of CBM BASIC. The added features include auto line numbering, tracing, single stepping through programs, line renumbering, and definition of keys as BASIC keywords. POWER even includes new "stick-on" keycap labels. The cursor movement keys are enhanced by the addition of auto-repeat and text searching functions are added to help ease program modification. Cursor UP and cursor DOWN produce **previous** and next lines of source code. COMPLETE BASIC program listings in memory can be displayed on the screen and scrolled in either direction. POWER is a must for every serious CBM user.

Call us today, for the name of the Professional Software dealer nearest you.

Professional Software Inc.
166 Crescent Road
Needham, MA 02194
Tel: (617) 444-5224 Telex #951579

™POWER is a registered trademark of Professional Software, Inc.

Record 1st Quarter Sales, Net Income, Earnings per Share are Announced by Commodore International Limited

Commodore International Limited has announced record sales, net income, and earnings per share for the first quarter of fiscal 1982 which ended Sept. 30, 1981.

Irving Gould, chairman of the board of Commodore, commenting on the record results achieved in the most recent quarter, noted that he was ''extremely pleased with the first quarter's record results, the quarter just reported,'' especially in light of the fact that the quarter ''is historically the slowest of the year for Commodore.''

Continuing his comments, Gould said that ''current demand for Commodore products is at an all-time high, and we expect the current quarter to be another record period. We also look for our current year's sales, net income, and earnings per share to be substantially above the record results registered in fiscal 1981 which ended June 30, 1981.''■

	1st Quarter (ended Sept. 30, 1981)	
	1981 (Unaudited)	**1980**
Sales	$54,150,000	$35,212,000
Income before Income Taxes	9,070,000	5,749,000
Provision for Income Taxes	1,790,000	1,232,000
Net Income before Extraordinary Item	$7,280,000	$4,517,000
Extraordinary Item—Tax Benefit of Net Operating Loss Carryforward	300,000	—
Net Income	$7,580,000	$4,517,000
Earnings Per Share:		
Earnings Per Share Before Extraordinary Item	$ .71	$ .44
Extraordinary Item	.03	—
	$ .74	$ .44
Average Shares Outstanding	$10,296,000	$10,311,000

BPI Business Accounting Systems Now Available from Commodore

The entire line of BPI business accounting systems, the most comprehensive series of accounting software available in the micro-computer industry, is now available from Commodore.

BPI Systems, Inc. has created the following accounting packages for Commodore: General Ledger Accounting, Accounts Receivable, Inventory Control, Job Cost, and Payroll.

The General Ledger Accounting System includes integrated accounts receivable, accounts payable, and all subsidiary ledgers for payroll accounting.

The Accounts Receivable System is for companies with larger portfolios of accounts receivable and for those that need statement preparation, aging reports, and extensive credit analysis.

The Inventory Control System is a perpetual inventory system by quantity and cost. It keeps you informed about all items in the inventory and tracks minimum quantities and back ordered merchandise.

The Job Cost System, which is for businesses that require individual job costing, handles as many as a hundred cost accounts for an unlimited number of jobs. The system helps prepare original bids for jobs, then keeps track of all labor and non-labor costs for each job.

The Payroll System manages payroll for any size company. It handles local, state, and federal payroll taxes, and subtracts up to six more customized deductions for each employee. Payroll checks are written automatically.

Each of these systems provide a variety of reports and can be merged with one another. Generally accepted principles of accounting have been used in the design of these five systems. Each is written entirely in the language of the business and professional community, and can be used immediately without prior knowledge of computers.■

Professional Business Software

For The Commodore 8000 Series Computer System

CMS GENERAL ACCOUNTING SYSTEM II:

A fully interactive General Accounting System designed especially for the first time user. All input requests are fully prompted with complete verification of input data. Most reports may be printed either to the screen or the printer and started or stopped at any point. The user is led completely through each function by a series of highlighted prompts fully explaining the required input at each point. A professionally written instruction manual is included which shows sample reports generated by the system and further explains each step and prompt as it is encountered by the user. These user prompts, together with the detailed step by step manual, make it virtually impossible for the user to accidentally crash the program or to get lost in the program and be unable to proceed or backup. Some of the many features of each of the four major accounting functions is shown below.

GENERAL LEDGER:

Up to a 1000 accounts on the Chart of Accounts. Fully departmentalized up to nine departments. Cash Disbursements and Cash Receipts Journal as well as a General Journal for ease of data entry. Maintains account balances for Present Month, Quarter to Date, and Year To Date. User customized financial statements. Accepts postings from Accounts Receivable, Accounts Payable, Payroll, or other programs.

ACCOUNTS RECEIVABLE:

Prints Invoices and Monthly Statements. The finance charge rate and period may be set by the user. Full invoice aging reports with aging breaks set by the user. During invoice data entry a copy of the Invoice is displayed on the screen and the information is typed in exactly as if the Invoice was in a typewriter. Accomodates full or partial invoice payments. Provides for Credit and Debit Memos as well as Invoices. Invoice File capacity is 2000 minus the number of customers multiplied by 1.4. Five hundred customers will allow room for 2100 invoices. Invoices may be distributed among up to nine different General Ledger accounts with automatic updating to the General Ledger.

ACCOUNTS PAYABLE:

Prints Accounts Payable checks with full check voucher detail for each Invoice paid. Prints detailed check register. Automatic application of Credit Memos. Complete invoice aging reports with aging breaks set by the user. Invoice File capacity is 2000 minus the number of vendors multiplied by two. Invoices may be distributed among up to nine different General Ledger accounts with automatic updating to the General Ledger Account File.

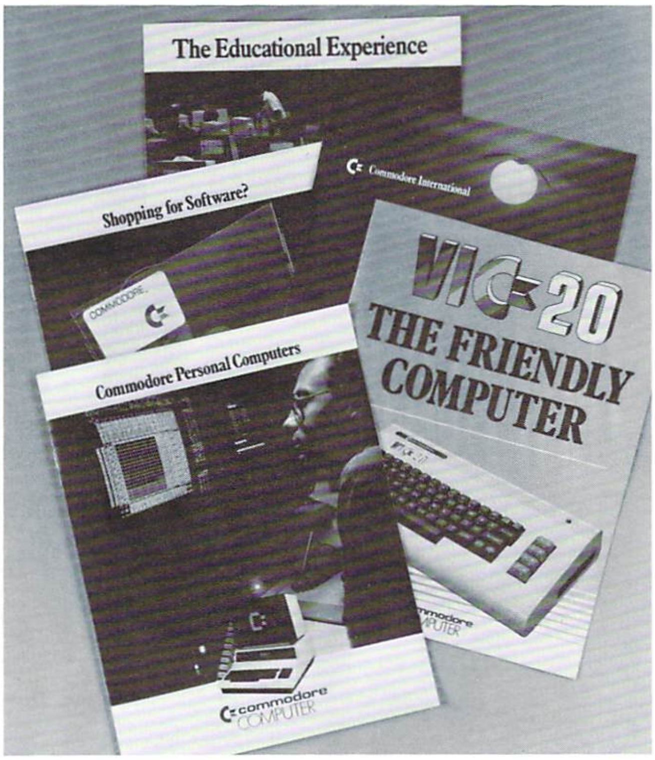
PAYROLL:

Maintains Monthly, Quarterly, and Yearly totals for each of up to 350 employees. Prints Payroll checks with full deduction and pay detail. Accomodates Weekly, Bi-weekly, Semi-Monthly, and Monthly employees. Pays regular, overtime, holiday, and piece work hours. Up to eight miscellaneous deductions or payments per employee. Prints Payroll Journal, Payroll Check Register, and an Absentee Report as well as 941 information and W2 forms. Automatic updating to the General Ledger.

See Your Nearest Commodore Dealer For A Demonstration

CMS Software Systems, Inc. 2204 Camp David, Mesquite, TX 75149 214-285-3581

New Product, Corporate Overview Brochures


The Educational Experience
Shopping for Software?
Commodore International
VIC 20 THE FRIENDLY COMPUTER
Commodore Personal Computers
commodore COMPUTER

Five new full-color brochures, which depict and describe the microcomputer and software product lines as well as the corporate capabilities of Commodore Business machines, Inc., are now available from authorized Commodore dealers from coast-to-coast for distribution to the computer-buying public.

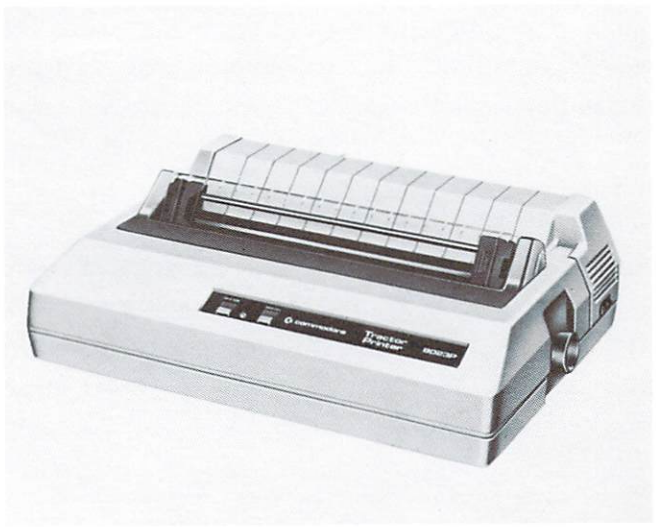
The product brochures, each eight pages, are: ''Commodore Personal Computers,'' which provides an overview of Commodore's four levels of microcomputers—the VIC 20™, PET®, CBM™, and SuperPET computer—as well as peripherals and software; ''The Educational Experience,'' which provides an insight as to how the Commodore PET has helped revolutionize classrooms around the world; ''Shopping for Software,'' providing a look at a wide variety of software systems including word processing, financial and information management systems, and systems tailored for specific industries, and ''VIC 20—The Friendly Computer.''

The VIC 20 brochure describes the many features of the exciting Commodore VIC 20.

Additionally, the VIC 20 brochure provides detailed information on the many new recreation and game programs, educational programs, home calculation, business and financial applications, and computing aids specially made for the VIC. Information and photographs of VIC system peripherals, including game controllers, plug-in program cartridges, Datassette tape recorder, VIC 1515 graphic printer, 3K, 8K, and 16K memory expanders, telephone modem, and others are also featured.

The new corporate overview brochure is a beautiful 16-page full-color insight to Commodore International Ltd., the parent company, and its subsidiaries, capabilities, and facilities. ■

New Low-cost, High-Speed Bi-directional Printer



The latest addition to the growing line of CBM peripherals is a bi-directional, 136-column printer with both tractor and friction feed. The 8023P is dot-matrix, and prints 150 characters-per-second (CPS). It is available through Commodore dealers throughout the nation for $995.00.

The new CBM printer is designed to operate through software control, prints upper and lower case alphabetic characters, all graphic characters available with a Commodore computer, as well as user-defined characters.

The 8023P conforms to IEEE interface requirements and connects directly to a Commodore computer. It is designed to be used with the CBM floppy disk drives, and may be daisy-chained with other IEEE-488 devices.

Because the printer is an''intelligent'' peripheral, it uses none of the computer's memory. In addition, the 8023P contains Random Access Memory (RAM), which permits storage of formatting data. ■

Commodore Announces Acquisition of Larger Facilities

Commodore has announced finalized plans for the acquisition of a 589,000-square foot building in the Valley Forge area for the manufacture of semiconductors, microcomputers, and related components, as well as administrative offices.

The building is at 1180 Church Road, Lansdale, PA. The purchase price is approximately $5.25-million for the building and some mechanical equipment. The closing date for actual purchase of the facilitiy will be during the first week of December.

In making the announcement of the acquisition of the building, Irving Gould, chairman of the board of Commodore International Ltd., Commodore's parent company, said that the company will expend approximately $15-million in renovating and equipping the building for Commodore's use.

"The growth of Commodore as a leading manufacturer of semiconductors and microcomputers has been both dramatic and substantial," said Gould, "and we see this new facility as the 'hub' of worldwide activities for both the parent company and our microcomputer subsidiary, Commodore Business Machines, Inc."

It is expected that the administrative offices of the Computer Systems Division of Commodore Business Machines, its Mid-Atlantic regional sales office, and Commodore International's corporate offices will be in the new facility by March, 1982. Commodore expects its MOS Technology subsidiary of Norristown, PA, manufacturers of microprocessors, to begin some operations in the building by late 1982.

The announcement was a follow-up to the announcement of the Pennsylvania Industrial Development Authority's approval of a $4-million loan to Commodore for the expansion program.

It was also announced that the expansion project should result in some 2,000 new jobs over a three-year period, which Pennsylvania Commerce Secretary Geoffrey Stengel, Jr., said was the biggest single job-producing expansion project involving state efforts in the past three years.

"We are very excited about the new facility and its virtually unlimited growth potential it affords us," Gould added. "Commodore is the only company in the microcomputer industry which manufactures its own microprocessors. Now we will be better able than ever before in our 23-year history to continue our role as an industry trend-setter in terms of technological advancement." ■

Six New Superpet Books

Six new reference books are now available for the SuperPET. The books are provided with the SuperPET system, but can also be purchased separately.

The SuperPET offers expanded capabilities by providing 96K RAM, an additional microprocessor, five languages, and a standard data communications interface.

The books include a System Overview of the SuperPET, and one book for each of the five languages available with the product—Waterloo microAPL, Waterloo microBASIC, Waterloo microFORTRAN, Waterloo microPASCAL, and Waterloo 6809 Assembler.

The System Overview book provides an introduction to the hardware of the SuperPET, an overview of the Waterloo software for the computer, and various descriptions that apply to the Waterloo software systems in general. The book retails for $5.95.

The Waterloo microAPL book, which retails for $9.95, is a tutorial introduction to the language, as well as a comprehensive reference manual.

The reference series also includes a Waterloo microBASIC book, which is divided into four parts: an introduction to the general characteristics of the system; a comprehensive reference guide describing the command language; an additional reference guide describing the programming language; and appendices containing summaries of both command and programming languages, as well as describing use of files with Waterloo microBASIC. It retails for $10.95.

The Waterloo microPascal book, also retailing for $10.95, features a tutorial introduction of Pascal language, and a reference manual defining the language.

The Waterloo microFORTRAN book is also divided into tutorial and reference sections, and retails for $10.95.

The final book in the series, pertaining to the Waterloo 6809 Assembler, describes the 6809 Assembler, Linker, and Monitor systems. It contains all the details necessary to develop and debug programs written in the 6809 assembly language for the SuperPET. Retail price is $10.95.

An additional book for COBOL will be available in the first quarter of 1982. These books may be ordered through Commodore dealers nationwide. ■

Programmable Character Set/ Gamegraphics Editor Now Available for VIC 20

A programmable character set and gamegraphics editor on cassette is now available for the VIC 20™ personal computer.

Now being sold at authorized Commodore dealers throughout the nation for $14.95, the character set editor comes with a 16-page instruction manual and allows VIC users to create groups of 64, 128, or 192 programmable characters at a time and use them in BASIC programs. Each group of characters takes only one-half kilobyte (0.5K) of program space.

With the new character editor, VIC 20 users can create their own characer set and easily modify letters, numbers, and graphics to include foreign language letters, mathematic and scientific symbols, or special "arcade" game graphics.

Commodore's new character set editor also allows VIC 20 users to save their newly-created character set on tape or disk for future use, and then easily insert the set in a BASIC program.

See the VIC section for further details.

Welcome Aboard!

Congratulations to the following groups, who have joined the lengthening ranks of dealers selling Commodore computer products. . .

Poston Computers
1008 34th St.
Bakersfield, CA 93301
805-324-4797

Compusolv
30 Lafayette Sq.
Vernon, CT 06066
203-871-7214

The Computer Center, Inc.
302 Commercial St.
Waterloo, IA 50701
319-232-9504

Computer Service of Shreveport
4436-A Youree Dr.
Shreveport, LA 71105
318-865-7189

Time Trend, Inc.
4432 Jackson St.
Alexandria, LA 71301
318-445-2627

Interstate School Supply
1835 River Rd. N.
Baton Rouge, LA 70821
504-387-5131

Audio Plus
111 Rena Dr.
Lafayette, LA 70503
318-988-2478

Tri-State Computers
1504 S. Salisbury Blvd.
Salisbury, MD 21801
301-742-2020

Mississippi School Supply
4515 Industrial Drive
Jackson, MS 39209
601-948-8600

Portsmouth Computer Center
31 Raynes Ave.
Portsmouth, NH 03801
603-431-7438

Computerability, Inc.
294 Main St.
Hackensack, NJ 07601
201-488-0990

Micro Computer Services
Pheasant Run Plaza
Warren, NJ 07060

Triton Data Systems, Inc.
23 Memory Lane
Denville, NJ 07834
201-627-9380

Mannfred Electronics Corp.
60-10 Cassena Blvd.
Flushing, NY 11355
212-762-7777

Computerland
6181 Jericho Turnpike
Commack, NY 11725
516-499-4484

Computerland
2258 Wyoming Mall
Albequerque, NM 87112
505-294-2900

Micro Computer Center
7900 Paragon Rd.
Dayton, OH 45459
513-435-9355

Controller Services, Inc.
2152 Mountainview Ave.
State College, PA 16801
814-234-8555

CN Formation
516 Huron Ave.
Pittsburgh, PA 18936
313-484-4390

Audio Mart
518 Fifth Ave.
New Brighton, PA 15066
412-846-3000

Ficomp, Inc.
109 Witmer Rd.
Horsham, PA 19044
215-441-8600

Professional Computer Systems
10520 Plano Rd. #206
Dallas, TX 75238
214-343-1328

Business Arts Corp.
3427 Roseland St.
Houston, TX 77006
713-524-1406

Micro Sales & Supply
2205-A5 West Division
Arlington, TX 76016
817-265-4670

Audio Fidelity
7217 West Broad St.
Richmond, VA 23229
804-285-8781

Compu Prose, Inc.
550 Eagan St. #203
Charleston, WV 25301

THE COMMODORE COMPUTER

"WHY SPEND $10,000? COMMODORE GIVES YOU WORD PROCESSING PLUS COMPUTING FROM $3,400."*

— William Shatner

You know your office is drowning in paperwork. Letters! Reports! Records! Word processing is the obvious answer. And now, Commodore makes word processing affordable for even the smallest office.

All you need is the Commodore CBM™8032 desk-top computer system plus one of our word processing programs. For a modest investment, you get all the features of a sophisticated word processor. You can insert, delete, re-arrange paragraphs, revise as many times as necessary. With no time wasted in hand-typing multiple drafts. Our word processing system displays 80 characters on a line. Text appears on the screen exactly as it will appear in print. And Commodore's screen-editor is the easiest, most versatile to use, with full cursor control.

But that's only the beginning. Because the Commodore CBM is a versatile business computer that can save you more time, work and money than you ever thought possible. Your can use it for Financial Planning, Data Management and Retrieval, Direct Mail Lists, Legal and Medical Accounting and Portfolio Management. And there are programs for Accounts Receivable and Payable. Inventory Control. Payroll. Engineering. Personnel Recordkeeping. And Tax Preparation.

COMPARE THE ALTERNATIVES AND YOU'LL SEE WHY COMMODORE IS THE COMPUTER TO BUY.

	CBM 8032
Price: Under $3,400*	✓
Computer Memory (RAM + ROM): Total = 50K	✓
80 Characters Per Line	✓
Built-in Green Display Screen	✓
Separate Numeric Keypad	✓
Standard Upper & Lower Case Characters	✓
5¼" Dual Disk Storage Capacity — 1,000,000 Characters	✓
National & Local Service/Support	✓
Over 250,000 Desk-Top Computers Sold World-Wide	✓
Over 500 Software Packages	✓
Over 23 Years in Technology	✓

*This chart is based on our standard CBM 8032 Business Series, plus an 8050 Dual Disk Drive. A wide range of printers are available, depending upon your requirements. Commodore offers several Word Processing Packages.

WE WROTE THE BOOK ON SOFTWARE

COMMODORE SOFTWARE ENCYCLOPEDIA

The Commodore Encyclopedia is a directory of over 500 software programs for business, education, recreation and personal use. Pick up a copy at your local Commodore dealer.

FULL SERVICE, FULL SUPPORT

Commodore dealers throughout the country offer prompt local service. In addition, our new national service contract with TRW provides nationwide support.

Visit your Commodore dealer for a hands-on demonstration of our truly affordable computers that do word processing and so much more.

*This price was configured on our CBM 8032 Business Series and 8050 Dual Disk Drive. A full line of Commodore printers is available, depending upon your requirements.

commodore COMPUTER

The Information Solution To Your Office Productivity Bottleneck!

Now . . . turn your Commodore computer into a powerful business system with Info-2001™ Software Applications.

Info-2001™ Software Applications are designed by expert consultants to meet the needs of the professional practitioner and businessman. All systems have been thoroughly field-tested at many business sites to insure complete functionality.

Professional Solutions!

For the attorney, consultant, engineer, architect, and other professionals . . . a Time Management/Client Billing System with these advanced features:

- Timekeeping and Disbursement Accounting - to keep track of time and expenses by client and matter.
- Client Billing - to print computerized bills on professional billheads.
- Accounts Receivable - to provide Aged Receivables reports for improved "cash flow" control.

The "parameter-driven" design allows you to customize the system to conform to the specific practices of your office. You get the best of both worlds—a "field-proven" packaged system (at off-the-shelf prices) with the "flexibility" to meet your specific needs.

Business Solutions!

For the small to medium size business... a comprehensive Management Accounting System with these modules:

- Sales Order Entry
- Inventory Management
- Accounts Receivable
- Accounts Payable
- Payroll
- General Ledger

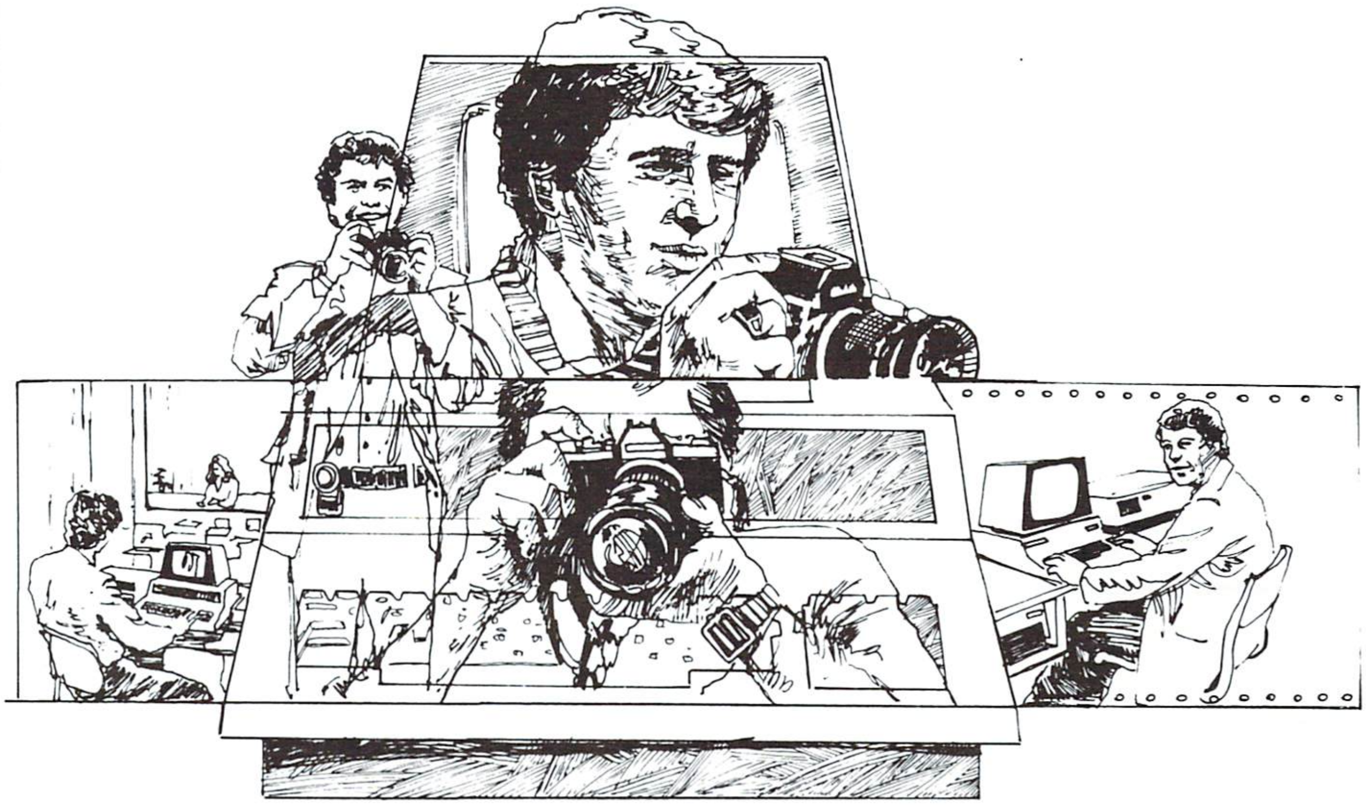
All software includes an instructionally designed Operator's Manual, functional input forms, and training by your authorized dealer using a computer-simulation model.

For a demonstration of these "Information Solutions" see your local authorized Commodore dealer today!

InfoDesigns

INFORMATION SOLUTIONS USING MICROCOMPUTER TECHNOLOGY

Info-Designs, Inc. • P.O. Box 5340 • W. Bloomfield, Michigan • (313) 582-9090



Pro Photographer 'Develops' Efficient Office Procedures Using CBM 8032

Much has changed in the field of photography since Matthew Brady toured the battlefields of the American Civil War, using revolutionary techniques to produce some of the world's earliest and finest photographs. Now, a century and a half later, photography has become a highly sophisticated field with an end-product that has a technical and creative worth limited only by the cameraperson and his or her equipment.

And while many of these "Matthew Bradys" of the 80's live in their darkrooms each weekend of the year, there are those who regard photography as much more than a hobby—the professional photographer whose livelihood depends on the ability to capture images through a lens and make a living doing so. While many of these professional picture-takers enjoy the envious role of making a living at something they enjoy, they are nonetheless business persons who must still pay salaries, purchase equipment and supplies, rent office space, pay taxes, insure the tools of their trade, and above all, show a profit at year's end.

To deal with the dreaded volumes of paperwork, Kevin Raber, a Villanova, PA photographer, has added an additional piece of equipment to his arsenal of cameras and lenses. While it uses no film, this newly acquired equipment does provide a clear picture of the financial and administrative status of his business—it is a Commodore CBM 8032 computer. Along with Commodore's 8050 disk drive and 4022 printer, and the most recent developments in accounting and information management software, Raber has transformed his successful business into a streamlined operation, void of troublesome paperwork.

Clean and uncluttered is the best way to describe the studio and offices of Raber Photography. Located in a fashionable suburban community northwest of Philadelphia, Raber's studio reflects the professional reputation he has required in the six years since he began a fledgling business in the basement of his home. Portraits of many of his customers, including first lady Nancy Reagan, are carefully arranged along the walls of his studio. It is obvious when you walk through his establishment that Raber is a well-organized individual who takes pride in his business. And he is obviously also concerned with keeping his paperwork in order—bills, invoices, statements, and customer orders continue to grow in proportion to his successful business.

Before acquiring his CBM 8032, Raber was becoming increasingly concerned with the management of this information. "We looked at all our paperwork and decided to find a permanent solution," Raber explained. "We saw that our statements, which were being done by an outside service, were returned to us in computerized formats."

'Now I have more time for photography and less headaches."

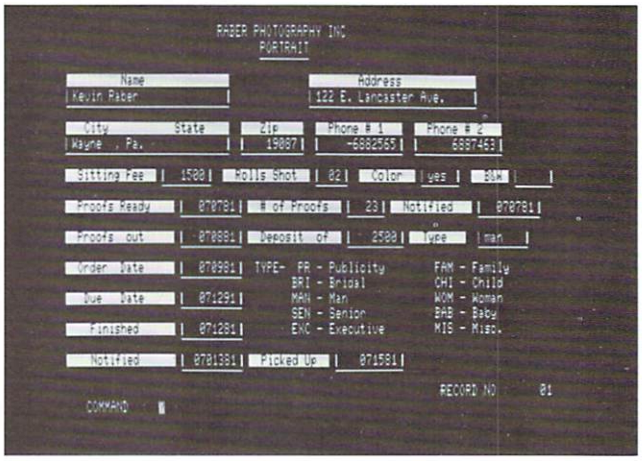
Raber's interest in purchasing a computer increased when he realized that although the information he was receiving from the service was fairly accurate, he was really doing all the work. "We had to give them all the bottom line figures and still keep track of the information," he said. "Plus," he added, "we were waiting three to four weeks to get our statements back—or have them 'developed' if you will!" Raber was also frustrated with the transposition errors over which he had no control.

Finally, he went shopping. Among his requirements for a computerized system were the ability to keep track of all his jobs at one central location, maintain and generate a list for direct mail advertising, and easily handle accounting and invoicing. Raber stressed the "easy" requirement because he had no accounting knowledge and wanted a system that would do all the work.

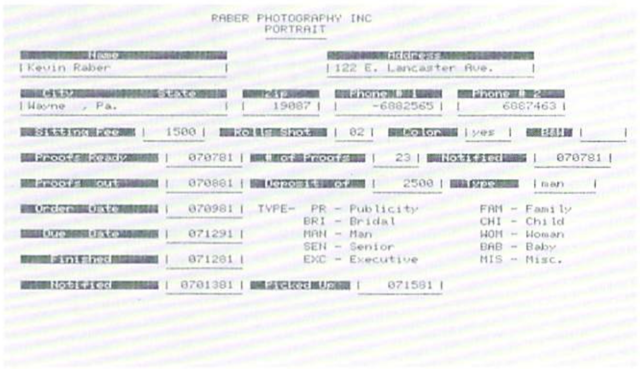
For several weeks, he closely evaluated all of the leading micros. "We made a list of the items we liked about each computer. Commodore came out on top because of the disk storage and also because of its ease of operation."

Another plus for the Commodore computer, according to Raber, was that "it looked like it was built like a tank." So far, Raber has seen nothing to change that opinion. "After three months we have not had an ounce of problems with it and it's very simple to operate," he said.

Again, an easy to use system was important to Raber because it would be used not only by himself, but also by his colleagues, who include his mother and partner Barbara Raber, associate Dave Campli, and secretary Julie McWilliams.



```
                    RABER PHOTOGRAPHY INC
                          PORTRAIT

     Name                               Address
|Kevin Raber              |       |122 E. Lancaster Ave.   |

 City            State     Zip       Phone # 1      Phone # 2
|Wayne  , Pa.            | |  19087| |  -6882565| |   6887463|

Sitting Fee |  1500| Rolls Shot | 02| Color |yes | B&W |    |

Proofs Ready |  070781| # of Proofs |  23| Notified |  070781|

Proofs  out  |  070881| Deposit  of |  2500| Type  |man  |

Order  Date  |  070981| TYPE-  PR - Publicity       FAM - Family
                               BRI - Bridal          CHI - Child
Due   Date   |  071291|        MAN - Man             WOM - Woman
                               SEN - Senior          BAB - Baby
Finished     |  071281|        EXC - Executive       MIS - Misc.

Notified     | 0701381| Picked Up |  071581|

                                               RECORD NO      01
   COMMAND   █
```



```
                    RABER PHOTOGRAPHY INC
                          PORTRAIT

     Name                               Address
|Kevin Raber              |       |122 E. Lancaster Ave.   |

 City            State     Zip       Phone # 1      Phone # 2
|Wayne  , Pa.            | |  19087| |  -6882565| |   6887463|

Sitting Fee |  1500| Rolls Shot | 02| Color |yes | B&W |    |

Proofs Ready |  070781| # of Proofs |  23| Notified |  070781|

Proofs  out  |  070881| Deposit  of |  2500| Type  |man  |

Order  Date  |  070981| TYPE-  PR - Publicity       FAM - Family
                               BRI - Bridal          CHI - Child
Due   Date   |  071291|        MAN - Man             WOM - Woman
                               SEN - Senior          BAB - Baby
Finished     |  071281|        EXC - Executive       MIS - Misc.

Notified     | 0701381| Picked Up |  071581|
```

Raber uses OZZ to generate his own internal forms, which are identical to the screen formats created by the system.

THE FRAMEWORK

Like many professional photographers, Raber divides his business into three major areas: portraits, commercial, and weddings. Additionally, there are miscellaneous jobs that are handled daily, such as making prints for customers. For each of these areas Raber used a separate form. Items such as wedding contracts, and portrait and long commercial forms used to be his only source of information.

Raber explained that he tried, without success, to consolidate these forms so they were always in the same spot. "Unfortunately," Raber conceded, "it never worked out that way."

Raber's studio is set up in such a way that the office area is far removed from the processing department, located downstairs in the building. Raber described the daily problem of trying to find a form for a particular job in the office when the paperwork was downstairs with the job being processed. "There were times when we didn't even know if a job was being worked on," Raber reflected.

Raber set out to alleviate this paperwork and information logjam by organizing his records using OZZ, Commodore's information retrieval system. Able to tailor the system to his own specific needs, Raber set up a similar format for each of the three main components of his business. With commercial accounts, for example, he established a Commercial Customer file, which keeps track of commercial customer information such as name, address, etc. In addition, Raber created a commercial customer job file, which keeps track of the client's specific job. "Now, it becomes very easy to keep track of the three or four separate jobs that we may be doing for the same client," he said.

A Price Quote file now makes life a lot easier when it comes time to bill a client. Raber, describing the benefits of the price quote file, explained, "We can arrange a special price for a customer. Instead of writing the quote on a piece of paper and losing it two months from now when we actually do the job, we now have the exact price quote information in the computer and we just call it up."

The same type of files have been organized for the commercial, portrait, and wedding areas of Raber's business. With weddings, for example, OZZ keeps total track of the payments and progress of the order, from the time the film gets sent out to be proofed until the order goes out for the album.

All areas of the business are tied into an invoice program, which automatically writes an invoice each time a job is entered into the system. Thus, if a customer calls on the phone to inquire about the financial status of an account, the current information is readily available. "The program will even calculate the tax and deduct the deposit, so the customer can be told the exact amount of the bill," said Raber.

"Not only does the Commodore computer and all the software save me time, it gives me an added edge on expanding my business."

Because everyone in the office uses the system, Raber has a smoothly run operation on his hands. "If Dave is handling a certain job, he will make any adjustments that are required. For example, he may update an account to indicate that the proofs are ready to be picked up. Julie may come in here the next day and actually notify the customer that the proofs are ready. When the customer has been reached, she will then amend the record to indicate that the customer has been notified. The system always contains the current status of each job."

Raber also uses OZZ to generate his own internal forms, which are identical to the screen formats created with OZZ. The proper form accompanies each job through each stage of work that is done. The form is amended to reflect the job's current status. This same information is then entered into the system in the identical format. Raber describes his computer as 'a gigantic filing system.' "We can sit here and call up any job we want at any time we desire. Before, we had a piece of paper in this room and a piece of paper in that room. We were relying on paper. And we had a very bad habit of not completing the information. Our CBM has trained us to become more organized," said Raber.

"The system keeps track of each and every step. Before, we sent a job down to the lab with a piece of paper for processing. Here in the office, we never knew the status of the job unless we ran downstairs and waded through the work being done. Now, we have all that information right here in front of us."

Raber also has several print formats stored in the system, giving him a hard copy of various aspects of his business. A list of all portrait jobs lets Raber know at a glance what jobs are completed. Raber can then contact customers who have failed to pick up their order. A commercial customer job list prints the commercial customer and the date an order is due. Raber prints this list once a week to let his staff know what jobs are due for completion.

MAILING LIST

Because he is constantly adding new customers, at a rate of ten to 20 per day, Raber is also compiling these names to create a comprehensive mailing list. Using Dr. Daley's mailing list software, Raber is developing specialized lists to cater to all aspects of his business. He is currently working on a list of senior high school students who will be having photographs taken for graduation. "This is a very competitive business," said Raber, "so being able to generate these specialized lists is one way to stay ahead of the competition."

DEBITS AND CREDITS

The biggest time saver of all for Raber has been the accounting system he implemented on his 8032. "I was never an accountant," Raber admitted, "and the only thing I've learned since I started this business is the difference between a debit and a credit. First, my accountant handled all the information. Now everything is taken care of by the system." The system Raber depends on is an accounting package developed by CMS Software Systems of Mesquite,Texas.

Upon initial installation of CMS, Raber ran parallel books with his accountant for two months to ensure the accuracy

→

SUCCESSFUL TEAM: Kevin Raber, mother and partner Barbara Raber, and their Commodore system all contribute to a smooth running business operation.

of the accounting package. Satisfied, Raber now explains that "my accountant doesn't have to run my statements every month. He just takes a look at the statements generated by CMS and makes his recommendations."

Raber has recognized an obvious savings in both time and money with his accounting system. "I don't have anybody doing bookkeeping anymore," Raber gladly pointed out, "so that's a $90.00 savings every month. I still pay my accountant's fee, but he's now getting accurate information, which benefits me when he completes my tax forms and advises me on business transanctions. Plus, my time is worth a lot of money and I used to spend it doing paperwork, rather than taking pictures. Now, I have more time for photography and less headaches.

"Before we purchased our Commodore computer, I used to get ulcers over the accounting. As the business began to grow, I would go home every night and worry. I would spend an entire day looking for $1.29.

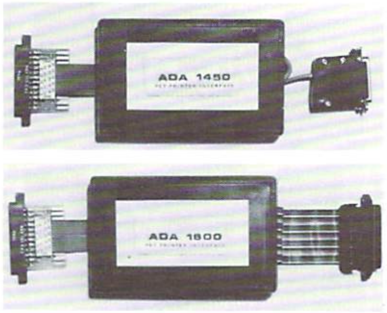
"Now, the system won't let me go out of bounds. Checks are automatically printed out. Dollar totals are debited and credited to the right account. Payroll checks are automatically calculated with full deductions and pay detail.

"The system gives me everything I need from an accountant except the advice. I don't think you can ever replace an accountant, because he can tell me what to do with the information. But I know he is reaching his conclusions using reliable and accurate data. All we have to do is put the numbers in the right place.

"In the beginning, I was very concerned about the success of buying a computer. Now I trust it and would be lost without it. Not only does the Commodore computer and all the software save me time," Raber concluded, "it gives me an added edge on expanding my business." ■

——Paul Fleming

CBM/PET INTERFACES



RS-232 SERIAL PRINTER INTERFACE – addressable – baud rates to 9600 – switch selectable upper/lower, lower/upper case – works with WORDPRO, BASIC and other software – includes case and power supply.
MODEL – ADA1450 149.00

CENTRONICS/NEC PARALLEL INTERFACE - addressable – high speed – switch selectable upper/lower, lower/upper case – works with WORDPRO, BASIC and other software – has Centronics 36 pin ribbon connector at end of cable.
MODEL – ADA1600 129.00

CENTRONICS 730/737 PARALLEL INTERFACE – as above but with Centronics card edge connector at end of cable.
MODEL – ADA730 129.00

COMMUNICATIONS INTERFACE WITH SERIAL AND PARALLEL PORTS – addressable – software driven – true ASCII conversion – selectable reversal of upper-lower case – baud rates to 9600 – half or full duplex – X-ON, X-OFF – selectable carriage return delay – 32 character buffer – centronics compatible — much more.
MODEL – SADI 295.00

ANALOG TO DIGITAL CONVERTER – 16 channels – 0 to 5.12 volt input voltage range – resolution is 20 millivolts per count – conversion time is less than 100 microseconds per channel.
MODEL – PETSET1 295.00

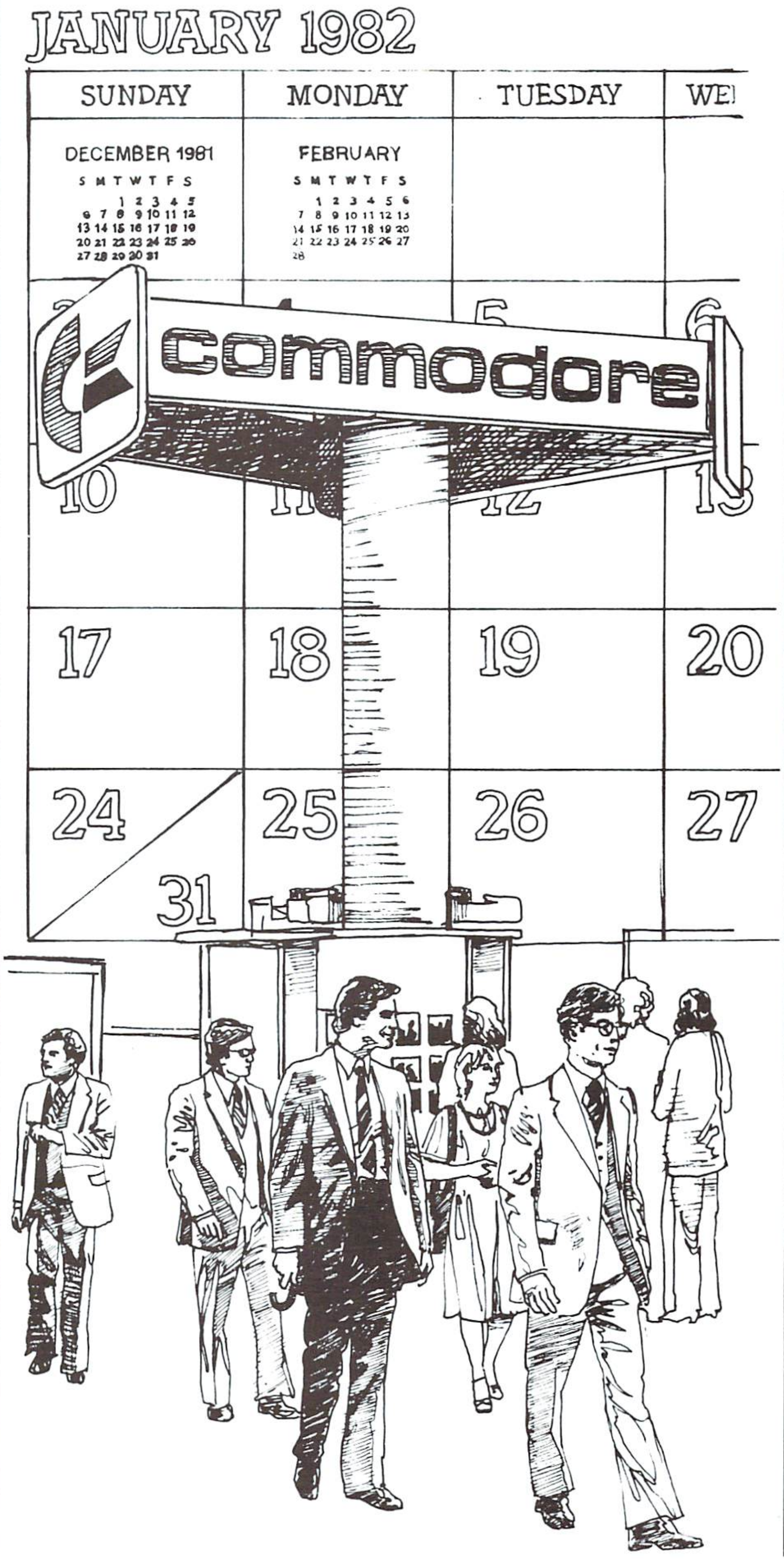
REMOTE CONTROLLER WITH CLOCK/CALENDAR – controls up to 256 devices using the BSR X10 remote control receivers – 8 digital inputs, TTL levels or switch closure — 8 digital outputs, TTL levels.
MODEL – PETSET2 295.00

All prices are in US dollars for 120VAC.
Prices on 220 VAC slightly higher.
Allow $5.00 shipping & handling, foreign orders add 10% for AIR postage.
Connecticut residents add 7½% sales tax.
All prices and specifications subject to change without notice.
Our 30 day money back trial period applies.
MASTER CHARGE/VISA accepted.
MENTION THIS MAGAZINE WITH YOUR ORDER AND DEDUCT 5% FROM TOTAL.

IN CANADA order from: Batteries Included, Ltd., 71 McCaul Street, F6 Toronto, Canada M5T2X1, (416)596-1405.
IN THE USA order from your local dealer or direct: Connecticut microComputer, Inc., 34 Del Mar Drive, Brookfield, CT 06804, (203)775-4595.
Dealer inquiries invited.

CmC

Connecticut microComputer, Inc.
34 Del Mar Drive, Brookfield, CT 06804
203 775-4595 TWX: 710 456-0052



Commodore's National Trade Show Schedule

Throughout 1982, Commodore will display its expanding product line to the public through various trade shows and exhibitions. The turnout and participation at these events is always encouraging, as thousands of present and future Commodore enthusiasts get a first-hand look at the latest hardware, software, peripheral, and applications products.

If you're a veteran of these events, or plan to attend for the very first time, here is a convenient list of the major national shows at which Commodore will be represented in 1982.

January:

7-10	Consumer Electronics Show Las Vegas, Nevada

February:

22-24	Federal DP Expo Washington, DC

March:

to be announced	AMDA St. Thomas, Virgin Islands

June:

6-9	Consumer Electronics Show Chicago, Illinois
7-10	NCC '82 Houston, Texas
28-30	COMDEX-Spring '82 Atlantic City, NJ

July:

15-17	NOMDA '82 Kansas City, Missouri

September:

22-24	Federal Computer Conference Washington, DC

October:

11-14	INFO '82 New York City

November:

27-30	COMDEX-Fall '82 Las Vegas, Nevada

JINSAM

Data Manager selected by NASA, Kennedy Space Center

SAVE TIME! SAVE MONEY!

JINSAM saves taxpayers 100's of thousands of $

OFFICIALLY APPROVED BY COMMODORE

"Much more powerful than you can imagine!"

Robert Baker
Kilobaud Microcomputing

JINSAM is an integrated system. It makes it easy to use your information to its fullest. No more will hundreds of valuable hours be spent searching or analyzing needed information nor re-entering information for various reports.

JINSAM transforms your desk-top computer into the "state of the art" data processing machine with features and accessories found nowhere, even at 10 times the price. NASA, Kennedy Space Center selected JINSAM 8.0 and saved approximately $95,000 over other software/hardware costs. Riley County, Kansas also selected JINSAM 8.0 and saved approximately $90,000 over other software/hardware costs.

JINSAM is designed for you. It is forgiving. It has help commands for every option, available at the touch of a button. The amount of information you store, its structure and/or your hardware can change but your data won't have to be re-entered. Recovery utilites are included even for catastrophes, security passwords are built in for privacy, simple editing and entry includes auto recall, and deleting records is easy and the space is reclaimed. JINSAM includes TWO FREE accessories for reports and labels. You have unlimited report formats with summing and lined up decimals and the label printer prints up to 5 across – any size and even allows single envelopes or index cards.

JINSAM 1.0 allows fast and easy file handling, manipulation and report generation for any CBM computer with CBM 2040 disk drive. It features a menu for ease, has encrypted passwords, 3 deep sorts, .5 to 3 second recall.

JINSAM 4.0 for CBM 4000 series adds JINSORT, user accessible machine sort of 1000 records in 15 seconds; compaction/expansion of information, automatic list maintainance, unlimited number of fields, unlimited record length and much more.

JINSAM 8.0 for CBM 8000 series has all 4.0 features plus unlimited sort, horizontal format, and search by key or record number.

JINSAM 8.2, NEW FOR '82 expands 8.0 capabilities by adding information search by word, key or record number and machine language print, format and manipulation routines.

★ CUSTOM DATA FILES
★ CUSTOM REPORTS/LABELS
★ KEYED RANDOM ACCESS
★ FAST/EASY/MENU DRIVEN
★ MULTIPLE SEARCH KEYS
★ PRIVACY ACCESS CODES
★ WILD CARD SEARCH

JINSAM EXECUTIVE version (soon to be released) is our most powerful professional system for the CBM 8000 and 9000 series. Executive will have 8.2 extended features plus allow multiple users with in-use lockout protection, executive command files, automatic math relations, join, merge or link files, greatly increased record capacity and machine information search by word, as well as by key or record number and many, many more features.

There are currently 7 more interfacing modules – and more under development, including independent interfaces between JINSAM and business packages for your G/L, A/R, A/P needs. We announce the availability of modules and enhancements in JINSAM's quarterly newsletter.

WORDPROPACK - Intelligent interface for WordPro 3, 3+, 4, 4+, creates lists of information from JINSAM files. It allows up to 10 conditions based on each item of information. Produce individualized letters, report cards, special reports, checks, invoices, etc.

MULTI-LABEL - Prints multiple labels per record with up to 2 lines for messages and consecutive numbering. Produce inventory, caution labels, bulk mail labels, etc.

MATHPACK - global calculator/editor +, −, *, /, by another field or constant; null (remove contents) of a field or replace contents of a field with any word, number or phrase. Sum multiple fields in each record or running sum of single field in all records. Extract information or effect permanent change. Replace in the same field or place in a waiting field.

DESCRIPTIVE STATPACK - Determine MEAN, MEDIAN, MODE, STANDARD DEVIATION, VARIANCE, RANGE. Generate HISTOGRAMS from 1 to 25 steps, and produce Z-SCORE reports.

ADVANCED STATPACK - (You must also acquire DESCRIPTIVE STATPACK) Generate one, two or three way CROSSTABS (number of occurance) CHI SQUARE, LINEAR REGRESSION, with graphic representation and prediction, LINEAR CORRELATION and SIMPLE ANALYSIS OF VARIANCE.

CALCPACK - 2 way interface to VisiCalc or any user program. It lets you use VisiCalc for complex manipulation, editing, placing results in JINSAM for sorting, storing or moving data to WordPro as well as giving the ability for exchange with your own applications.

INTERAC - Interface which can read VisiCalc files, WordPro files and almost any sequential files to build JINSAM databases automatically. For example: You could "download" information on holerith cards to sequential files and INTERAC would place them into JINSAM files.

All accessories are accessed thru the JINSAM menu and require security password to gain entrance.

JINSAM gives you FREEDOM OF CHOICE. Start with JINSAM 1.0 and upgrade hardware and data at any time. Choose from accessories at any time. The JINSAM Newsletter brings the latest updates, user input and uses and keeps an eye on the future.

JINSAM stands alone by placing "a lot of potential computing power in one integrated program package" (Fred Klein, Ferson div. of Bausch & Lomb). "The JINSAM package is justification for buying a system no matter what the hardware, be it Vector or Commodore or whatever the system" (Larry Colvin, Micro Computer Systems). It is a "grandfather" in this young field since the first JINI MICRO database system has been marketed since 1979. JINSAM EXECUTIVE will be the third generation in development. All JINSAM systems are sophisticated and flexible yet easy to use. JINSAM is saving its users valuable time and money in government, educational and research institutions, business and industry worldwide.

JINSAM is a Commodore approved product. See your local dealer for a demonstration.

JINSAM is a trademark of JINI MICRO-SYSTEMS, Inc.
WordPro is a trademark of Professional Software, Inc.
VisiCalc is a trademark of Personal Software, Inc.
CBM is a trademark of Commodore Business Machines, Inc.

JINSAM Data Manager

. . . . Additional Information and nearest dealer
. . . . Newsletter Subscription ($5 US/$8 Worldwide)
. . . . JINSAM Demo Disk ($15, plus shipping & tax)
. . . . User's Guide ($40.00, plus shipping & tax)

Please send to:

Name ______________________

Position ______________________

Company ______________________

Address ______________________

City, State, Zip ______________________

Phone ( ) ______________________

Computer, Disk ______________________

Application ______________________

JINI MICRO-SYSTEMS, Inc.
Box 274-CI • Kingsbridge Station
Riverdale, New York 10463
(212) 796-6200

Dealer Inquiry Welcome

Compu-Mania

California Computer School Caters to all Ages

Everyone knows that Commodore computers can do many amazing things but now they are even getting families together. That's right, at CompuKids, a computer school in Los Angeles, through a program called CompuFamily, parents and children are getting together to learn about computers using Commodore PETs.

As the name implies, CompuKids was created to teach children about computers through game and learning activities. Created by Dr. Julie Chan and Ellen Newman, CompuKids is so successful that the school is now expanding into new areas. Besides CompuKids and CompuFamily, the school now offers: CompuTeens, for ages 13 and up, CompuFolks, for adults, and even CompuKinder, for four and five year olds. Eventually Dr. Chan and Ms. Newman plan to expand the staff, but for now it is just the two of them.

Some of the Compukids pause to pose with their PETs.

Dr. Chan says that the decision to choose the Commodore PET was made when her partner was researching microcomputers for a PTA committee. After many hours of research Ms. Newman found, like so many others before her, that Commodore computers give the most computing for the money. Specifically, she found that the PETs featured "intelligent" peripherals and they were virtually maintenance free. Ms. Newman had heard that "Commodore made the most reliable of all the microcomputers," and said that she "found it to be true." She has had no major problem with the PETs even though "over 200 children have been pounding away at them."

One of the features that Dr. Chan says she likes best in the PET is that "editing is terrific and less complicated than the TRS 80." She also likes the fact that the PET requires less cables than Apple and has the built in green screen with large letters. Dr. Chan also said that she loves the graphics because "they are so easy compared to other computers."

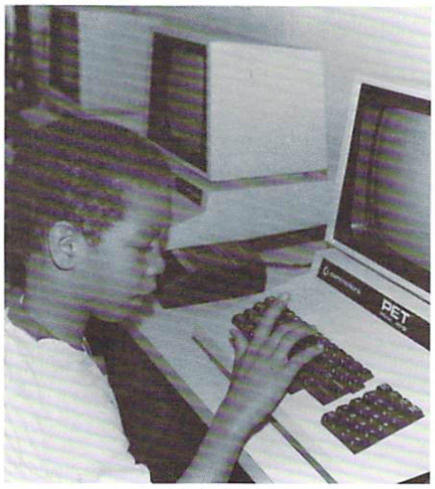
Aside from all the different CompuKids courses taught by Dr. Chan and Ms. Newman they have now also started teacher seminars to familiarize teachers with computers.

Long range plans also include the possibility of writing a basic programming textbook for children and perhaps even becoming a Commodore dealer, because so many of the parents want to buy computers for their children after they have taken the course.

One thing that is helping the school expand is the heavy publicity that the school has received. The school has been described in newspapers and magazines, including People Magazine, and has been featured by TV stations as far away as Japan. The CompuFamily course has been very popular and has received the most media coverage. This has caused some "pleasant" problems because the class has been interrupted so often by television crews and reporters.

Computers can sometimes intimidate people who have never used them. Adults seem to have more trouble with the idea of using a computer than children, but, after a course like CompuKids the result for both adult and child alike seems to be the same. Dr. Chan tells of one woman who came to the class tense and literally shaking. The woman wanted to make sure that she could get her money back if she didn't like the course. By the end of the day the woman felt so confident on her PET that she didin't want to leave! Another story she tells is of one little girl who didn't want to leave the class

→



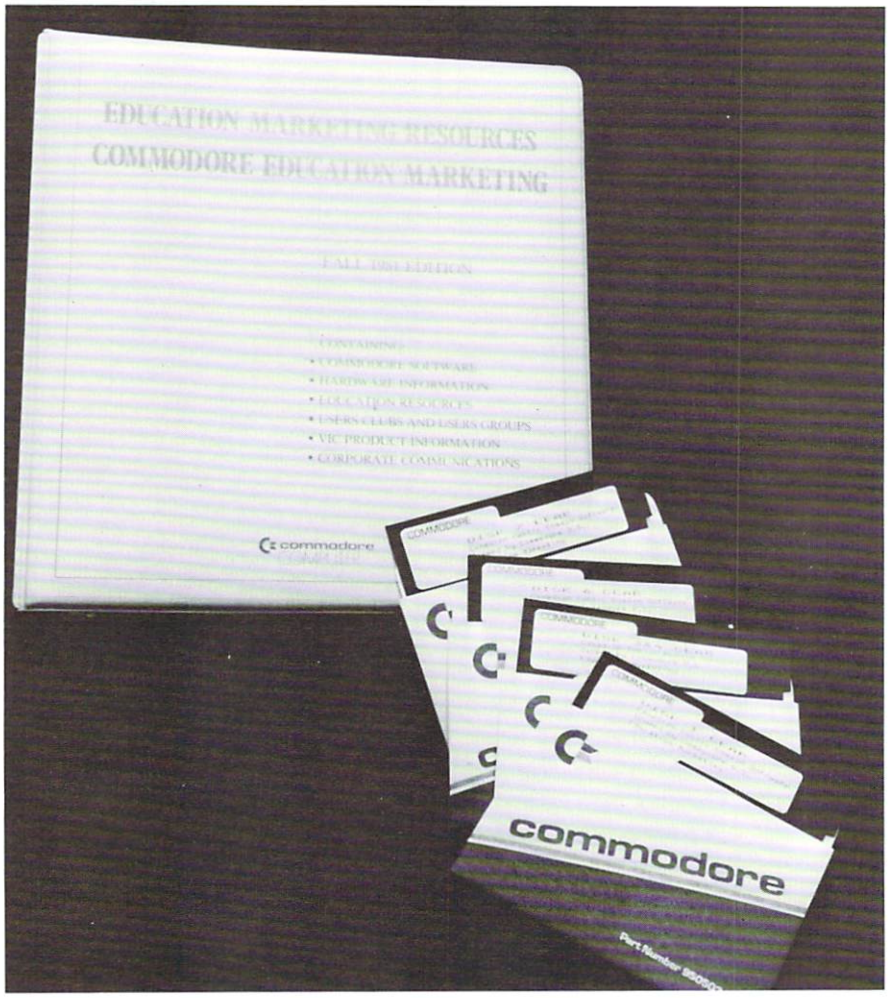
at the end of the day, claiming the computer would keep her company at night.

Through Chan and Newman's teaching skills and user friendly Commodore PETs, children, teens, adults and even whole families are becoming confident and comfortable with computers. Chan feels that this is vitally important, especially for children.

"In the coming years, computer literacy will be as important a skill as reading and writing are to us now," she said "and it is vital that learning occurs now while they are young." ■

—John O'Brien

Education Marketing Resources Book



The Education Marketing Resources book (U.S. Fall 1981 Edition) is now available from Commodore on a limited basis. This edition was designed to assist dealers, school administrators, teachers and students. Contributions to this volume have been submitted by people all over the United States and Canada.

Some of the highlights of the book are:

- 185 Public Domain software programs on four disks.
- Lists of User Clubs and software vendors.
- 380 pages of materials to support educators.
- VIC product information.
- New product spread sheets.
- Special Education Resource Center materials.

The public domain disks included in the book may be copied freely. The retail price of this book is $24.95 and is available from Commodore dealers. ■

Stephen Baccus— "Just a 12 year-old kid who learns faster"

Some people have pet cats or dogs. Others have more exotic pets like tarantulas or skunks. Stephen Baccus, however,is both the rule and the exception, because his pet is a boa constrictor named Julius Squeezer!

But there's another disparity between this 12 year-old and other pet owners. You see, Stephen Baccus also has a very special electronic pet—a PET computer. And he's even programmed it to sing.

More than that, Stephen also creates other programs on his PET, most of them problem-solving. Unusual for a 12 year-old. But Stephen's true uniqueness lies in his genius. He has an IQ of 190, and, when he took his Scholastic Aptitude Test (the college entrance exam), he scored 1420! A perfect score is 1600. A child prodigy? Yes. A pain-in-the-neck brat? No.

Stephen is not the type to brag about his IQ, nor is he the type to brag about is acceptance last Fall into New York University on a full academic scholarship. There, he is currently studying the rather unusual combination of dramatic arts and computer science. "Drama and entertaining are my first love, I guess," he admits, "and computers is a back-up field."

"He was ready when he was 10," said Stephen's father, James. "But you just don't send a kid to college at 10. I decided to let him 'mature' a little."

Stephen taught himself to read before he was two with help from the children's television show, "Sesame Street." He was permitted to skip school at the age of eight, but grew bored and began sitting in on high school classes. He took algebra and was—you guessed it— an A student.

Stephens's love of dramatics is evidenced by his extensive portfolio of acting experiences: he's appeared in the Jerry Lewis movie, "Hardly Working" (in which, according to Stephen, he plays the role of a "brat"), he had a part in a Florida production of "The King and I," and, right now, he's working on a few TV and radio commercials. And he's also developed a one-man, one-hour long act which includes singing, dancing, comedy, and magic tricks which he performs at local Manhattan nightspots.

He is quick to explain that his schedule does leave "some time" for his school work. And when he's not busy with academics or dramatics, programming his PET computer is a favorite pastime. According to his mother, Florence, "Computers take up a large part of his day, but he has other interests." He also likes reading scientific magazines and watching old movies on TV. He has a passion for flying and has even passed a flying test. But he can't get a pilot's license until he's 16 and he can't fly solo because he needs a flight instructor to assist him. At 4-foot-9, his feet don't reach the pedals.

12 Year-Old College Freshman Stephan Baccus and his PET Computer. Stephen is Currently Attending New York University.

So Stephen is a well-rounded individual. But he is not, his mother insists, a "walking encyclopedia." He has the ability to learn quickly, Mrs. Baccus said, "But that doesn't mean he knows that many facts. If you give him a date, he doesn't necessarily know what happened on that date."

"At first, when people met me, they're in awe," said Stephen. "Then they realize I'm just a 12 year-old kid who learns faster."

Sure, just a 12 year-old with a pet snake, a PET computer, and the whole world ahead of him. ■

—Jody Miller

REQUEST™

A SELF-COMPILING
RELATIONAL DATABASE

DATABASE SELECTION—

- Uses screen masks to form query
- Provides extensive search capabilities
- Search arguments can include arithmetic/boolean functions, multi-field comparisons
- Queries can generate input for automatic database maintenance
- Queries can be stored in "Query Library" and executed from menu on demand
- Any number of fields can be queried concurrently
- Query output can be routed to disk, CRT report formatter, VISICALC™ or SOURCE™

DATABASE MAINTENANCE—

- Uses sophisticated screen formatting & data entry, like on IBM 3270's!
- Generates it's own screens automatically!
- Handles records up to 4K in length, using multiple screen "Pages"!
- Automatic data compression for increased disk capacity
- Uses SuperKram access method for incredibly fast access, LESS THAN .2 SECONDS FOR A RECORD!
- Automatic index creation/maintenance
- Automatic maintenance capabilities
- "Goof-Proof" error handling
- Input can come from VISICALC™ or SOURCE™

DATABASE REPORTING—

- Automatic headlines
- Automatic field editing
- Report fields can be calculated, sub-totaled & cross-footed in any manner desired
- Optional counter breaks may be set
- Automatic grand totals
- Automatic statistics

REQUIREMENTS—

SuperKram (available separately) and: Commodore Pet 32K (40 or 80 col.) and 2040/4040/8050 disk.

ONLY $225

USA UNITED SOFTWARE OF AMERICA

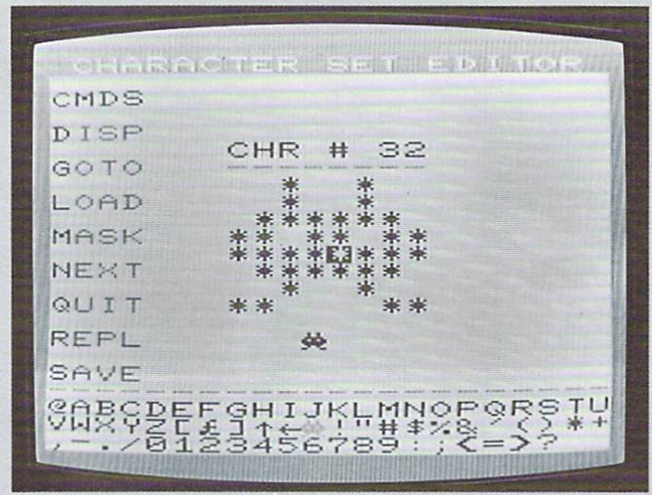
750 THIRD AVE.
NEW YORK NY 10017

Look for the RED-WHITE-BLUE United Software Display at your local computer dealer, or send check or moneyorder, plus $3.00 shipping

(212) 682-0347 Telex 640055

First in a Series:

Using the Programmable Character Set Editor



We wanted to tell you about a fascinating new product for the VIC 20 and the best way to do that is to give you an example of what you can do with the product. THE TUMBLER is a screen animation program which you can type directly into your VIC 20 and RUN.

You can use the editor to write the SAME type of program yourself, LIST, SAVE and RUN it in BASIC. Try typing TUMBLER exactly as shown below, and type RUN. You can also SAVE this on tape for future use by using your Commodore DATASSETTE data recorder, or on disk using the VIC 1540 DISK DRIVE (available January).

```
10  REM:TUMBLER BY M. TOMCZYK
11  K=7168:FORJ=32*8TO32*8TO32*8+7:
    POKE7168+J,0:NEXTJ
12  READ CH:IFCH=-1THENGOTO20
14  POKEK,CH:K=K+1:GOTO12
20  PRINTCHR$(147):X=64:POKE36869,255
25  PRINT:PRINTSPC(6)CHR$(18)CHR$(30)
    "TUMBLER!"CHR$(146)CHR$(31)
30  POKE36878,15:36875:B=128:E=138:PRINTCHR$
    (31)
40  FORM=BTOESTEP5:NEXTM:POKES,M:POKES,
    0
50  PRINTTAB(80-X)CHR$(X)CHR$(32)CHR$(145)
60  FORM=BTOESTEP5:NEXTM:POKES,M:POKES,
    0
70  IFX=  79THENED
80  FORT=1TO150:NEXTT
90  X=X+1:B=B+5:E=E+5:GOTO40
100 FORP=1TO100:NEXTP:GOTO20
500 REM:TUMBLER CHARACTERS
505 DATA24,24,127,24,56,40,40,40,
510 DATA96,96,60,50,240,72,36,18
515 DATA96,98,60,16,114,78,66,192
520 DATA50,52,152,113,50,44,32,24
525 DATA0,4,104,113,62,48,73,134
530 DATA65,34,20,217,242,60,192,0
535 DATA65,34,148,89,50,252,192,0
540 DATA48,8,228,28,12,63,48,0
545 DATA64,32,16,240,16,56,84,68
550 DATA0,64,32,24,248,28,42,34
555 DATA1,2,4,254,30,40,72,132
560 DATA38,22,8,24,52,82,144,144
565 DATA12,76,56,14,8,24,36,34
570 DATA48,178,84,56,16,16,40,68
575 DATA177,178,84,56,16,16,40,68
580 DATA48,48,16,56,84,16,16,48,-1
```

The key is the DATA starting on lines 505. What the Character Editor lets us do is this:

1. First we created a series of special characters using the Character Set Editor.
2. We then typed "SAVE" from the editor's command menu and typed "LIST." The list command gave us a list of 8-number DATA statements for each of the special characters we created. Each special character is represented by ONE LINE of DATA statements.
3. The BASIC program we then wrote includes a section which lets us identify our programmable characters as DATA statements, tells the VIC to READ that DATA and create characters in memory and on the screen (POKE 36869,255) and PRINT those special characters in line 50(the X is the special character).
   Immediately after we PRINT the character (CHR$(X)) we PRINT a space to erase that character (CHR$(32)) and tell the VIC to move one space to the left and PRINT the next character one space over. In this way we create the illusion of animation on the screen.
4. When you RUN the program, the programmable characters appear and move. If you want to experiment, you can change some of the numbers in the DATA statements and watch how the characters change.

The VIC Programmable Character Set & Gamegraphics Editor is a useful aid for both BASIC and Machine Language programmers. . .and is friendly enough for first-time computerists to use with little or no experience. The Character Editor retails for only $14.95 and is available from most VIC dealers. It comes with a 16 page instruction manual which talks you through all the features, step-by-step.

The Character Editor lets you perform all sorts of fascinating tricks with your VIC 20. . .but most importantly it lets you create your own characters! This means you cna create foreign language character sets, game graphics, special notation symbols. . .anything your imagination conjurs up. We'll have more ideas for using the Character Editor in upcoming installments in this series.

VIC=20
The friendly computer

For VIC/PET/CBM Users:
THE COMMODORE DATASETTE— SUCCESSFUL LOADING TECHNIQUES

For those of you who use a Commodore Datasette recorder, here are some helpful hints to keep those tapes rolling!

LOAD ERROR . . . a programmer's nightmare when you've just LOADed a program from your Commodore Datassette and this glaring message blinks up on your screen. What did you do wrong? Was it your program, the Datassette, the tape cassette, sunspots, poltergeists, or what?

There are several techniques which can help you load program tapes more reliably. Here are a few of them:

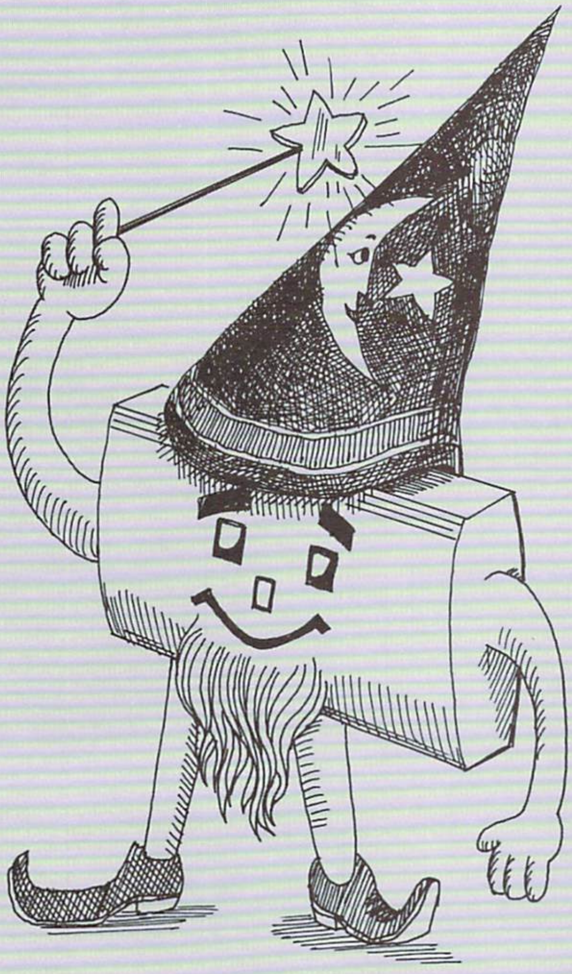
1. Tightening New Tapes
If you're using a new tape, either a blank tape or a pre-recorded tape (Commodore sells a variety of prerecorded tapes), it's always a good idea to FAST FORWARD the entire tape and then REWIND it before using it. This tightens the tape on the spool and reduces the possibility of load errors due to loose tape in the cassette.

2. Positioning The Datassette
Placing the Datassette too close to your television set may expose it to interference which can produce LOAD errors. Try placing the Datassette away from the television set, and avoid coiling the Datassette cord. Commodore tests show that Datassettes placed on top of a television set produce more LOAD errors than Datassettes positioned away from the set with the cord fully extended.

3. Save: Save
When SAVEing programs on tape, it's always a good idea to save it twice. You can do this in one step. For example, if your program name is "MAGICIAN," you should SAVE it twice by typing: SAVE "MAGICIAN": SAVE "MAGICIAN" and hit RETURN. These are actually two identical commands separated by a colon. After the first MAGICIAN is SAVEd, the second command is read and the program is recorded again. Pre-recorded tapes sold by Commodore are recorded a minimum of three times so if one program is damaged or if you get a LOAD error, you can type LOAD again and get the next program on the tape.

4. Cleaning and Realigning Tape Heads
Like any tape recorder, the Datassette requires servicing to clean and demagnetize the tape heads and to adjust the head alignment. If your Datassette has been in use for some time and is producing errors, it may require servicing.

These techniques should improve your Tape Loading techniques. . . whether you're using a VIC, PET or CBM with your Commodore DATASSETTE tape recorder. ■



The VIC Magician

by
Michael S. Tomczyk
VIC Product Manager

"LEARNING ABOUT THE CURSOR"

The subject of this "magical" installment is how to position or "program" information to print where you want it on the screen. This covers everything from how the cursor works to how to write programs that print words or graphics in specific locations on the screen.

Cursoring Around
Cursor control is one of the VIC 20's most powerful features. (The cursor is that blinking square on the screen that tells you where the next symbol will appear).

There are many ways to move the cursor around the screen, to make it appear and disappear and do crazy things . . . but the cursor's real power is its ability to *position* graphics, letters, numbers on the screen. Let's explore the cursor in depth and see how it works.

When you first turn on the computer, you should see a blinking blue rectangle (the "cursor") directly below the opening display. The cursor is controlled by the CRSR keys.

CRSR Keys
The CRSR keys are located at the lower righthand corner of your keyboard. They're the ones with the arrows on them. As you can see, these two keys let you move the cursor to the right or left, up or down. If you press the [ ] key, the cursor moves *down* the screen. If you hold down the SHIFT key and press the *same key* the cursor moves *up* the screen.

Moving the cursor right and left is just as easy. The [ ] key moves the cursor to the right and SHIFTing the same key moves the cursor to the left.

Special Features of the VIC 20 Cursor
Here are some special features of the VIC 20 cursor controls:

1. Automatic repeat. If you hold down either of the two CRSR keys you'll discover that the cursor automatically *repeats* as long as you hold it down. This is to help you move quickly to a desired location and is a powerful "screen editing" feature of the VIC 20.

2. Scrolling. If you press the "down" cursor key and keep holding it, you'll see that the cursor moves to the bottom of the screen . . . and when the cursor hits the bottom the entire screen will *scroll up one line at a time*. This is to give you more space when you're writing a program. Note that the VIC 20 screen only scrolls when you "cursor down." Moving the cursor to the top of the screen does not have any scrolling effect. Normally, scrolling only occurs when you move the cursor *down*.

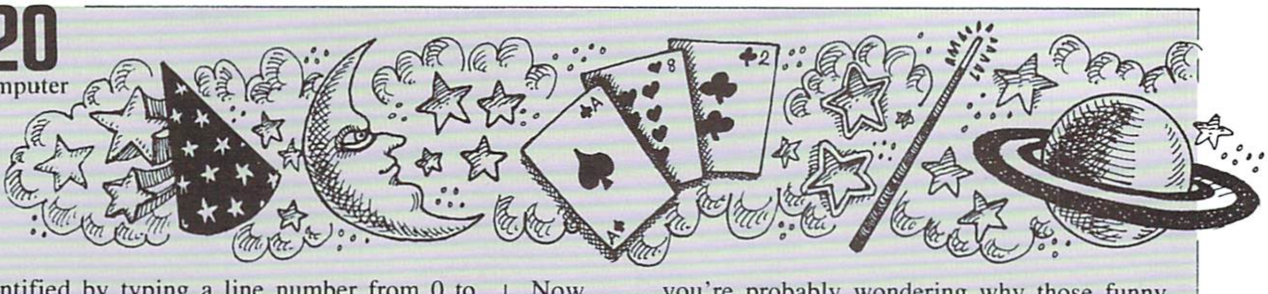
3. Wraparound. Try moving the cursor *horizontally* by pressing the CRSR RIGHT key. Notice that when it reaches the end of the line you're on, it automatically *jumps down to the beginning of the next line*. Conversely, if you SHIFT CURSOR LEFT the cursor will move left and jump *up* one line when it reaches the edge of the screen. This process of jumping up or down from one line to the next is called "wraparound."

4. CLR/HOME. Often, you want to move the cursor to the top lefthand corner of the screen. This is called the"home" position. The HOME key on the VIC is located at the top right corner of the keyboard. If you hit this key the cursor moves "home." If you hold down the SHIFT key and type CLR/HOME, you are actually typing *CLEAR* which *erases* any information you might have on the screen and positions the cursor at the home position.

To see how these keys work, try typing some information on the screen. Now type the HOME key. The cursor jumps to the "home" position. Now type CLEAR by holding down the SHIFT key and typing CLR/HOME. All information is gone and the cursor is in the 'home" position.

5. RETURN/SHIFT. You can also move the cursor down the lefthand column by hitting the RETURN key . . . but be careful when using this method. As a computer, the VIC has been taught to read and understand computer *programs*,

→



which are identified by typing a line number from 0 to 65000 in the far lefthand column of the screen. If you type a word without any line number and that word is not one of the "commands" in the VIC's vocabulary, the VIC will tell you that you've made a *programming error*. Here's how it works . . .

Try this: Hit the CLR/HOME key and type the word HELLO. Now hit the RETURN key. The VIC responds by telling you you've made a SYNTAX ERROR. This makes it difficult to type several lines on the screen. One way to overcome this is to type your message and then hold down the SHIFT key and hit the RETURN key. The cursor will move to the next line but the HELLO command will not be "entered" and the VIC will not give you an error message. To try it, type HELLO, then hold down the SHIFT key and hit RETURN.

This SHIFT/RETURN key combination can be a very powerful feature. If, for example, you're drawing a graphic picture . . . you can draw the picture and move to the next line quickly using SHIFT/RETURN without getting any error messages. This is helpful because a common technique in creating graphic programs in BASIC is to first draw the graphic picture on the screen, then add line numbers, quotation marks and the PRINT command along the left-hand column to convert your picture into a numbered BASIC program. (See the COLOR & GRAPHICS chapter in the VIC owner's manual). SHIFT/RETURN is also useful for moving around a BASIC program displayed on your screen when you want to move to different areas for editing purposes without affecting the program lines.

Programming the Cursor
So far, we've discussed some ways to move the cursor in *direct mode*. Now let's see how you can move and position the cursor in your computer *programs*.

You can PRINT cursor commands inside your computer programs—just like letters, numbers and graphic symbols. The format for doing this is exactly the same as PRINTing any VIC character. Try typing this program line:

```
10 PRINT "HELLO"
```

Now press the [CRSR RIGHT] key five times between the first quotation mark and the word HELLO. If you type this it will appear on your screen like this:

10 PRINT "]]]]] HELLO" (Don't forget to hit RETURN at the end of the line to enter it).

Don't worry about the reverse bracket signs. We'll explain those in a moment. Now type RUN and hit RETURN. In the previous example, the word HELLO was printed in the left column. Now the word is printed 5 spaces over to the right because we put five CURSOR RIGHT commands in our program. The VIC moved the cursor five spaces from the left column and printed the word HELLO, just as it was instructed.

Now . . . you're probably wondering why those funny brackets appeared when you typed the CURSOR RIGHT key. The VIC uses special reverse symbols to show you where cursor commands are located in your program. This is helpful when editing a program or studying a program you haven't seen before. Here's a list of the graphic symbols used to represent the various cursor keys:

CURSOR RIGHT
CURSOR LEFT
CURSOR UP
CURSOR DOWN
HOME
CLEAR

The key to positioning a word or graphic image somewhere on the screen . . . or even making it MOVE in animated programs . . . is using the CURSOR key with the PRINT statement. Here are some exercises to give you some practice:

Exercise 1.
Type the same HELLO program except use CURSOR UP instead of CURSOR DOWN. This is an interesting one.

Exercise 2. This is one of the most common programming techniques.

```
10 PRINT "[CLR/] HELLO"
20 FOR X=1to1000
30 PRINT "[CLR/] BYE"
```

Exercise 3. Try combining the CLEAR and CURSOR RIGHT commands.

```
10 PRINT" ]]]HELLO"
20 FOR X = 1 TO 1000
30 PRINT" ]]]]]]]]GOODBYE"
40 FOR X = 1 TO 1000
50 GOTO 10
```

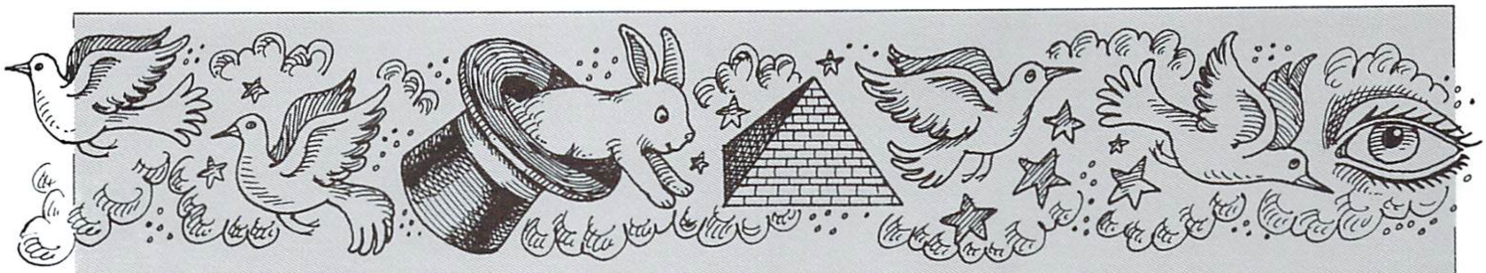
Exercise 4. Here's another version of Exercise 3 using just the CLEAR command.

```
10 PRINT"[CLRHME]HELLO"
20 FORX=1TO1000:NEXT
30 PRINT"[CLRHME]GOODBYE"
40 FORX=1TO1000:NEXT
50 GOTO10
```

Exercise 5. A simple animation example.

```
10 PRINT"CLRHME O":FORX=1TO150:NEXT
20 PRINT"CLRHME ]0":FORX=1TO150:NEXT
30 PRINT"CLRHME ]]00":FORX=1TO 150:NEXT
40 GOTO10
```

Moving the Cursor With CHR$ Codes
Having learned that the cursor can be included in PRINT statements like any VIC character, it stands to reason that cursor commands would have their own CHR$ codes so you can use them in CHR$ statements, like VIC characters.



The format CHR$(90) provides a more powerful alternative to the PRINT statement when displaying and manipulating VIC characters. All characters, including function keys and cursor controls, have their own CHR$ code numbers. For example, the number for the letter "Z" is 90, so if you type the following command, the letter Z will be printed on the screen:

```
10 PRINT CHR$(90)
```

This looks more clumsy than simply PRINTing the letter Z, but in many instances you can't or don't want to use the PRINT statement, so you use CHR$. In any event, here are the CHR$ numbers for the cursor controls:

CHR$ CODE	SCREEN MOVEMENT
29	CURSOR RIGHT
157	CURSOR LEFT
145	CURSOR UP
17	CURSOR DOWN
19	HOME
147	CLEAR

The format for using this technique in a program is to type:10 PRINT CHR$(29)

Exercise 6.
Here's an example of how you can print a "CLEAR" command using the CHR$ technique:

```
10 PRINT CHR$(147)"VIC CLEARS, MOVES HOME
   AND PRINTS MESSAGE"
20 PRINT CHR$(17)CHR$(17)CHR$(17)"NOW
   DOWN"CHR$(17)
30 PRINT CHR$(29)CHR$(29)CHR$(29)"THEN
   RIGHT"CHR$(17)"OKAY!"
```

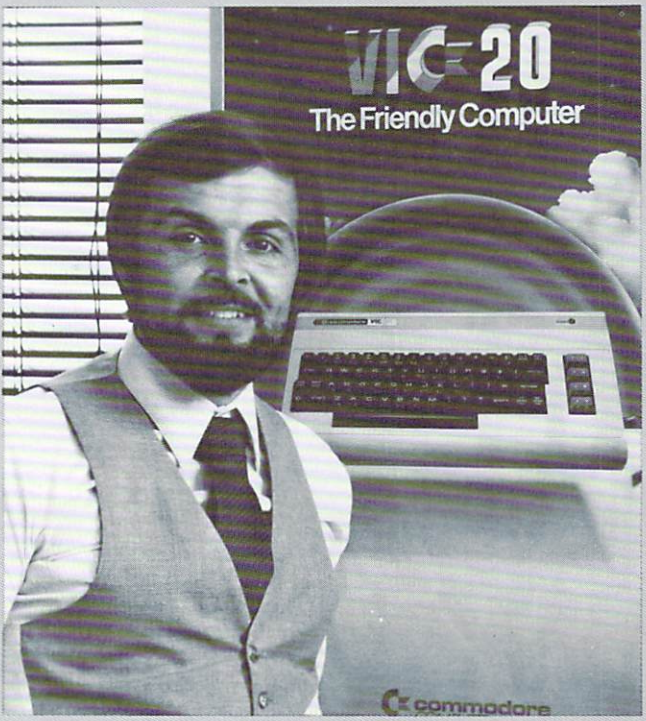
Things to note in this example include how the CHR$ statements are placed after the PRINT command. . . how in line 20 information printed in quotes can be mixed with CHR$ statements . . . how several CHR$ statements can be printed in a row to move the cursor more than once . . . how different messages and CHR$ statements can be "mixed and matched" as in line 30.

Exercise 7.
You can use "variables" in CHR$ statements, for example if you're going to be using the statements several times in a large program. Variables are important and we'll do a future article on them, but for now think of a variable as one or two letters which can be used as a *substitute* for a number, word, sentence or other piece of information. In this case, we will begin by "defining" our variable A equal to 147 (CHR$ code for CLEAR). This means we can substitute the letter A for the number 147 in my program. Note that we can still use the letter A normally in words and sentences, and that we can still use the number 147 if we want. This example simply clears the screen and prints HELLO.

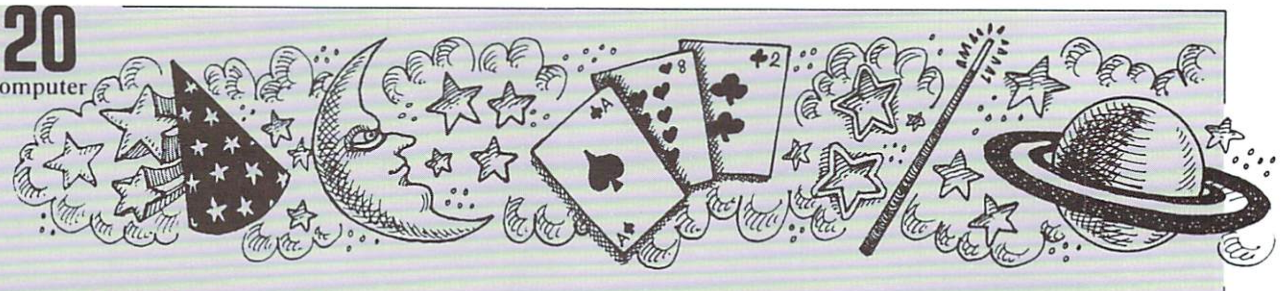
```
10 A=147
20 PRINTCHR$(A)"HELLO"
```

These few examples were designed to help you understand how the VIC 20's *screen editing* commands work, especially in your programs. It's one thing for a computer to be as flexible as the VIC in placing information on the screen, but it's equally impressive that you can *write these positioning commands in your BASIC programs!*

The VIC "MAGICIAN" is a continuing series of how-to articles designed to take new VIC owners several steps beyond the VIC 20 Owner's guide. If you would like to learn more about programming the VIC 20, Commodore provides several excellent learning tools . . . including the VIC 20 PROGRAMMER'S REFERENCE GUIDE and the TEACH YOURSELF PROGRAMMING SERIES of books and tapes. Commodore also provides several special "computing aid cartridges" including the VIC 20 SUPEREXPANDER CARTRIDGE, VIC PROGRAMMER'S AID CARTRIDGE, and VICMON machine language monitor. ■



Michael Tomczyk, VIC Product Manager



The VIC Magician

by Michael S. Tomczyk
VIC Product Manager

TIME DELAY LOOPS . . . AN ADVANCED TECHNIQUE MADE EASY

In this article we want to show you an advanced BASIC programming technique which first time computerists often stumble over . . . unnecessarily. If you're just starting out in computing it's important to remember that it's just as easy to learn a so-called "advanced" technique as it is to learn a "simple" one. The problem is that good descriptions of advanced tecniques are hard to find . . . it seems like you have to read through a whole book to reach them . . . or search through a dozen computer journals for an explanation you can understand.

We forged some new ground with our innovative "PERSONAL COMPUTING ON THE VIC 20" which comes free with every computer, but there are a lot of so-called "advanced" techniques which VIC owners are ready for as soon as they finish reading their owner's guide.

One place to get advanced programming information is the VIC 20 TEACH YOURSELF PROGRAMMING SERIES, which contains a "friendly" self-teaching programming manual and some interactive tapes which lead you through the lessons, step by step.

Another good source of programming information is the VIC 20 PROGRAMMER'S REFERENCE GUIDE, which every VIC owner should own. This invaluable "bible" of the VIC covers everything from the VIC's BASIC vocabulary to machine language programming tips.

Both the TEACH YOURSELF PROGRAMMING series and the PROGRAMMER'S REFERENCE GUIDE are available through your Commodore dealer.

The Time Delay Loop . . . Special Use of "For . . . Next"
One of the best "magic" tricks which programmers use to control the speed of their programs is called the "time delay loop." This is a simple line which you put in your BASIC program to make it move at a given speed. The technique is simple. All you do is include a line which says:

FOR T = 1 TO 1000:NEXT

You can include the line anywhere in your program, wherever you want a "time delay" and you can include several delays in different places if you want. For example, the first program below PRINTs two messages, separated by a "time delay."

The T in the time delay line can be any letter, two letters, or a letter and a number, but we usually use a T to specify "time" because FOR . . . NEXT loops can be used for purposes other than time delay. Also, using a T for time makes it easy to spot the time delay loops when you list a program with a lot of FOR . . . NEXT loops used for different purposes.

Another changeable item in the time delay loop is the number 1000. This can be *any number*. A larger number makes the time delay longer and a shorter number shortens the time delay. Actually, what you're doing is telling the VIC to count to 1000 (or whatever number) before proceeding. If the number is large, the VIC takes a longer time to count than if the number is short.

EXAMPLE 1 . . . TIME DELAY

```
10PRINT"THE VIC 20 IS GREAT!"
20FOR T = 1 TO 1000: NEXT
30GOTO10
```

This prints the message, "THE VIC 20 IS GREAT!," counts to 1000, and goes back to line 10 to print the message over again. The time delay specifies how long the VIC should wait before printing the message over again. Try substituting X or A2 for the "T" in line 20 and you'll see that this doesn't change the program. Try putting another number (200 or 2000) instead of 1000 in line 20 and see how the program gets faster or slower.

Time delay loops can be used to lengthen or shorten the duration of musical or sound effect tones being played on the VIC, as shown in the following example:

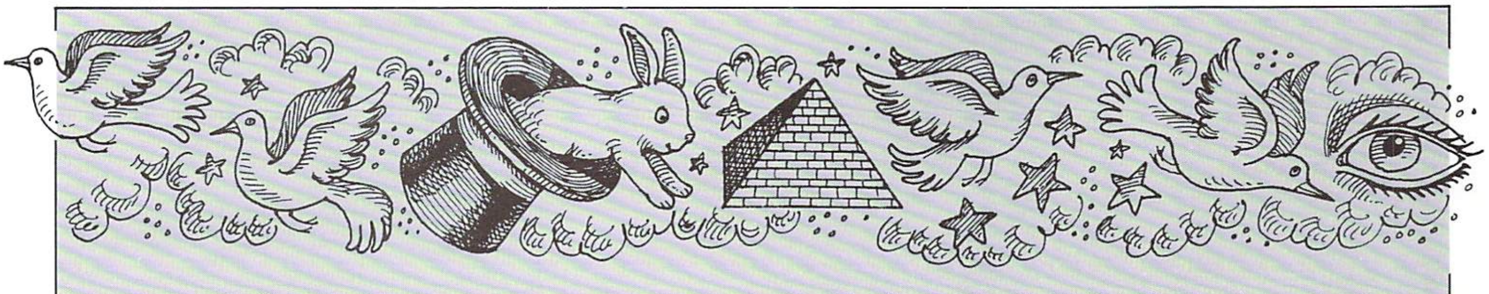
Example 2 . . . Time Delay Loops With Music
Here's a program which uses a time delay loop with a musical sound effect. The time delay relates to the length of time each note is played. You might pay special note to line 30 as well, which uses a FOR . . . NEXT . . . STEP statement to "step" through a range of VIC musical note values (from the Table of Musical Notes in the VIC user manual). Although we're looking mainly at time delay loops, there are other uses for the FOR . . . NEXT statement which we will cover in a future article. Back to time delays . . . here is the music program:

5 PRINT"WATER FILLING UP"
Prints this message on the screen while the sound effect is playing.

10 V=36878:S=36878
Set the volume equal to V and the speaker we want to use (in this case 367876) equal to S1.

20 POKEV, 15
Set the volume at maximum level (15)

30 FORN=195TO225 STEP 1:
VIC speakers can accept note values from 128 to 255. Here we are saying, for note values from 195 to 225, POKE Speaker 1 with those values, STEPping up one



at a time from note to note.

40 FORT=1TO100:NEXTT

This says count to 100 before moving to line 50 (where we play the next note). This is our TIME DELAY LOOP.

50 NEXTN

POKE the "NEXT N" into Speaker 1 . . . keep doing this until we reach the limit (which we set at 225 in line 30).

60 POKES, 0

Turn off the speaker, otherwise it will keep playing.

In this example, we see how a time delay loop affects the duration of a series of musical notes we want to play. If you want the notes to be shorter, change the "100" in line 40 to a smaller number. OR . . . use a larger number to make the notes play longer. Notice that if you make the notes VERY short (change 100 to 2 in line 40) you get interesting sound effects.

You can get "reverse" sound effects by changing line 30 to: 30 FORN=225TO195STEP−1:POKES,N This reverses the notes and steps backwards (−1) from 225 to 195 for a "water emptying" sound.

The key lesson here is the FOR . . . NEXT loop. In EXAMPLE 2 above we used the loop for two purposes . . . first, a time delay loop to extend the duration of notes we are playing and secondly, we used the FOR . . . NEXT loop to define N as a *series of note values*, then instructed the VIC to play that series one at a time by STEPping from value to value. The NEXT N in line 50 was the place where the note was actually played. You can prove that the note is actually played when the program hits line 50 by adding this line to your program:

45 PRINT"PLAY NOTE"

The program will now print "PLAY NOTE" just *before* it plays each note because you inserted the PRINT instruction before the NEXT N (next note) instruction in your program. This is important because it illustrates how you can combine sound effects with printed information (or graphic symbols) in your programs. Just mix and match PRINT statements with sound effects and you're writing programs that "sing."

Conclusion

We've taken a quick look at two major uses for time delay loops . . . one to place a "time delay" between two parts of a program to make it run slower . . . the other to place durations in a musical program where the time delay affects the sound being produced.

The best way to use time delay loops is to experiment. The best way to find out if time delays are required by your program is to see how fast it runs. If it runs too fast, slow it down by inserting a time delay. You can put a time delay loop anywhere you can put an ordinary program line, and you can use as many time delays as needed in a single program. You can even include a time delay as a GOSUB routine and keep coming back to it if you have a long-running program which requires the same delay to be repeated several times.

More information on time delays is available in the VIC owner's manual, the VIC 20 PROGRAMMER'S REFERENCE GUIDE, and most books on BASIC for the PET/CBM microcomputer.■

LEARN

Discover how easy it is for you to get useful results from your VIC.

Understanding Your VIC Volume 1: Basic programming uses a proven step-by-step approach to teach programming. It costs $11.95

A cassette tape with two demonstration programs from the book is available for $7.95. It will save you typing time and eliminate typing errors.

VIC Software

Easy to use program for you to create new characters for graphics or games. Cassette $9.95.

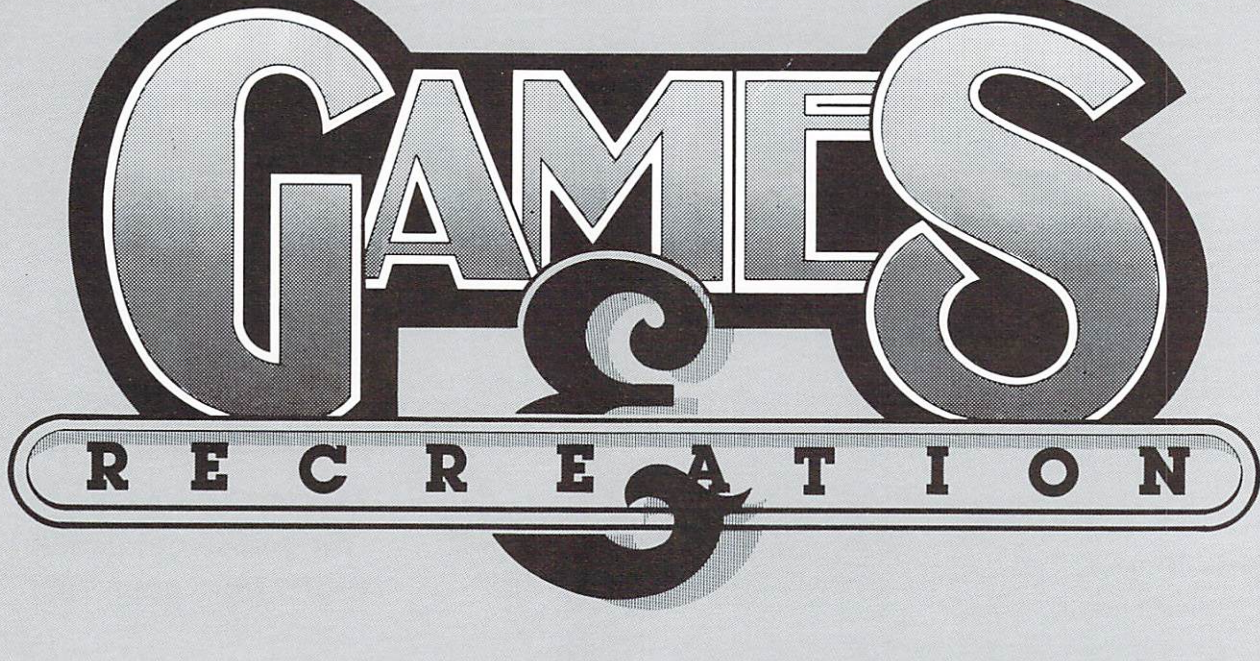
See your dealer or order direct
VISA/MC accepted
Money back guarantee
Please add $2 ($8 overseas) for shipping and handling

TIS INC.

Total Information Services, Inc.
Box 921, Dept. CM
Los Alamos, NM 87544

Dealer inquiries invited.

VIC=20
The friendly computer

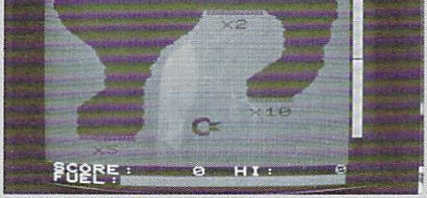


Christmas Special

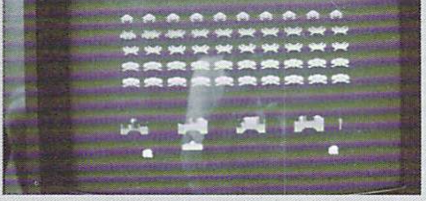
Cartridge Games for the VIC 20!

Wait till you see the new games from Commodore!

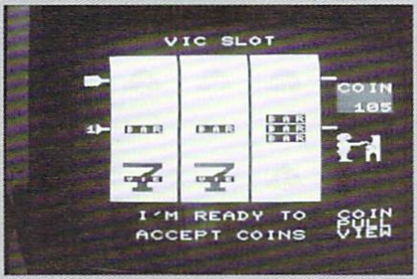
You'll see the difference between a true computer and a "video game machine" as soon as you see the first game. VIC 20 computer games have better graphics, resolution, sound effects and play action. Here's a quick look at the first games available for Christmas:



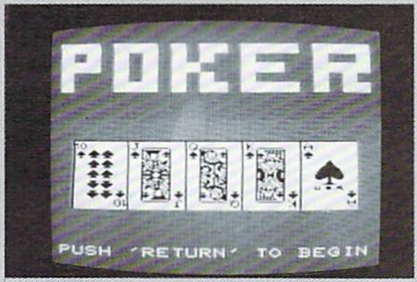
JUPITER LANDER . . . One of the most authentic landing games ever developed! Use your VIC keyboard like a control console to pilot your Jupiter Lander to a soft landing in one of the game's "caverns." Make as many exploratory landings as you can until your fuel runs out. Truly challenging! $29.95



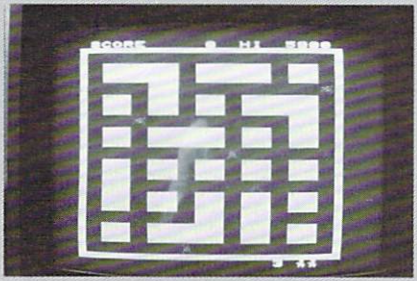
VIC AVENGER . . . This space action game has aliens, space cannons, flying saucers, and more! An "addicting" game for space-game enthusiasts. Commodore's most popular cartridge game! Joystick or keyboard. $29.95



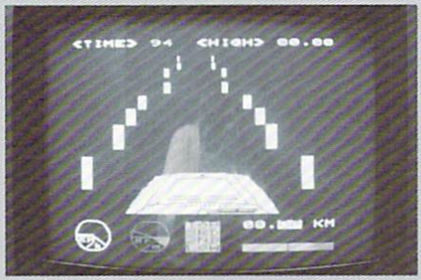
SUPERSLOT . . . Why lose money in Las Vegas or Atlantic City when you can have a computerized slot machine in your own home? Excellent animation and graphics, just like the computerized slots you see in casinos. $29.95



Draw Poker . . . This cartridge turns your VIC 20 into a "Poker Machine." You got to know when to fold 'em . . . but you only lose electric money when you practice your game on the VIC 20. Special high/low double draw feature. $29.95

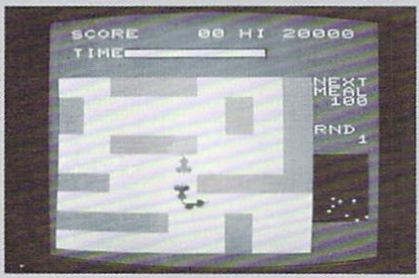


SUPER ALIEN . . . you're trapped in a maze . . . and so are the Super Aliens! You've got to capture all the aliens in the maze before they attack and eat (ugh!) you! A fast-paced game. Use joystick or keyboard. $29.95



MIDNIGHT DRIVE . . . This game turns your VIC keyboard into the dashboard of a race car. Play action combines road racing, time trials and night driving. Ignition, kilometers per second, rpm, 4 gears, accelerator, and more . . . it's all there. Fast-paced, authentic and challenging. $29.95

AND LOOK WHAT'S COMING SOON . . .
Here's a sneak preview of some of the games which are now in production:



RAT RACE . . . it's tough to be in a "rat race" if you're a real rat! With other rats chasing you, cats ready to pounce on you, and limited time in which to find and eat all the cheese. Several levels. Joystick/keyboard.

SCOTT ADAMS ADVENTURE GAMES . . . Previously available only on systems costing several thousand dollars, FIVE Scott Adams Adventure Games will be made available on cartridge for the VIC 20. These games include ADVENTURELAND, VOODOO CASTLE, PIRATE COVE, MISSION IMPOSSIBLE and THE COUNT. Adventure games are among the most popular games in the world, and are now being brought to VIC owners. The special features of VIC ADVENTURES is their ability to TALK! All 5 Adventure games are decoded to work with the VOTRAX (R) "TYPE 'N TALK"(tm) voice synthesizer which means if you attach a VOTRAX unit to the VIC 20, your Adventure games will SPEAK the words that appear on the screen!

SARGON II CHESS . . . called the best microcomputer chess game in the world, "SARGON II" will be offered on cartridge for the VIC 20. This is no ordinary chess game. it turns your VIC 20 into a real computer chess opponent. Difficulty levels for beginners through masters.

BALLY ... BALLY ... BALLY! ... Commodore's recently signed agreement with Bally means we'll be converting some of your favorite BALLY ARCADE (tm) and BALLY COIN-OPERATED GAMES to VIC cartridge! Exact titles and availability dates will be announced in the near future.■

VIC 20 COMPUTER GAMES ON CARTRIDGE

Commodore VIC 20 computer games on cartridge are just like real arcade games—not imitations. Judge the resolution, graphics, sound effects and play action for yourself. The difference between a real **computer game** and just a video game is easy to see. Here are a few reasons why VIC 20 computer games are best:

• **SCREEN POSITION:** When the display first appears on the screen, you can adjust the horizontal *position* of the picture by pressing the CRSR control key. This unique feature allows for variances between different television sets.

• **KEYBOARD/JOYSTICK:** Most VIC computer games work both from the keyboard *and* with a joystick. Most standard joysticks plug directly into the VIC 20 game port connection. (A few of our more sophisticated games/simulations use the keyboard—only as a "control console.")

• **SILENT DEMONSTRATION:** If you don't play a game within several seconds after turning it on, the game gives you a silent demonstration of itself to show you how the game is played! Also, the opening display on most games shows you which keys or joystick positions affect which actions.

• **HIGH SCORE CHALLENGE:** Another unique feature is the "HIGH SCORE" line which shows you the highest total so far. The VIC 20 "remembers" the highest score recorded . . . just like an arcade game . . . until you turn the VIC off. A few games (like RAT RACE) have preset high scores which give you targets to shoot for.

GETTING STARTED

1. Turn on your television set.
2. Turn your VIC 20 off (you will greatly increase the "life" of your game cartridges if you turn the VIC off before inserting or changing cartridges).
3. Insert the game cartridge.
4. Turn the VIC 20 on.
5. Adjust the picture on your screen by typing the CRSR key.
6. Type the appropriate START KEY. VIC games may be started by pressing one of the following keys: fl, P or RETURN. Here are some sample start keys:

TITLE	START KEY
VIC AVENGER	P
SUPERSLOT	P
JUPITER LANDER	fl
MIDNIGHT DRIVE	fl
SUPER ALIEN	fl
DRAW POKER	RETURN

7. Play the game using either the joystick or keyboard controls explained on the reverse side of this sheet.
8. Turn the computer off before inserting another cartridge.

INSTRUCTIONS

TITLE	OBJECT/RULES	SCORING	CONTROLS
VIC AVENGER (Arcade-style action.)	Use your laser cannon to score as many points as you can destroying the attacking aliens before they destroy you. Three rounds, aliens increase speed as they force skirmishes.	10, 20 or 30 points for aliens. Mystery points for flying saucers.	A Key=Fire L Key=Left ; Key=Right Joystick=Movement Button=Fire
JUPITER LANDER (A super space simulation!)	Land your spaceship safely on the only solid landing site on Jupiter. Make as many exploratory landings as possible before fuel runs out. Three landing sites (3 difficulty levels).	The softness of your landing is shown on the meters-per-second gauge on the right side of the screen. Land below the yellow zone=crash. The softer the landing the more points you get. Try to land with the marker high in the yellow zone.	A Key=Left thrust D Key=Right thrust f1 Key=Heavy thrust f2 Key=Middle thrust f3 Key=Low thrust (keyboard only)
SUPERSLOT (Casino-style action.)	Start with _____ coins—risk free—and bet up to 5 coins at a time. Just like Vegas and Atlantic City's computerized slot machines! Conserve coins, play the odds and try to increase your stake.	Win 2 to 3000 coins depending on the outcome of your slot machine "pull."	C Key=Drop coin V Key=View winning combinations P Key=Pull handle (joystick or keyboard) Joystick=Pull handle Button=Drop coin
DRAW POKER (Card play strategy game.)	Bet up to 9 coins. Draw 5 cards. Hold the ones you want to play, discard the rest. Bet again. Deal again—build your stakes. After you win you can draw 1 card (double or nothing).	Your hand and bet determines how much you win (or lose).	B Key=Bet C Key=Hold card RETURN Key=Hold card of your choice & proceed to next card. D Key=Double (keyboard only)
SUPER ALIEN	You have to clear a space maze of aliens, and your only weapon is an "alienbubble" to trap the aliens.	Blow up bubbles, trap the aliens, deflate the bubbles within a few seconds before the aliens eat their way out—and eat you! 100 to 1000 pts for each alien.	P Key=Move up L Key=Move left ; Key=Move right . Key=Move down A Key=Inflate D Key=Deflate Joy=movement Button/up=Inflate Button/down=Deflate
MIDNIGHT DRIVE	You're speeding down a darkened highway. The object is to drive as far as you can before your fuel runs out. Start (ignition), shift gears, steer, accelerate. Don't overheat or crash.	Drive for distance. Reach maximum KM in 100 seconds. Extra time for 6 KM.	A Key=Left D Key=Right f1 Key=Top gear f3 Key=3rd gear f5 Key=2nd gear f7 Key=1st gear RETURN=Acceleration I Key=Ignition (Keyboard only)
RAT RACE	You're a "rat" in a rat race, trying to find and eat all the cheeses before time runs out, an "enemy rat" catches you, or a black cat eats you!	2x cheese doubles score. Each cheese=100 and each extra cheese increases by 100. Challenge round (go for speed) 20,000 pt=Bonus Rat.	A Key=Magic stars P Key=up L Key=Left ; Key=Move right . Key=Move down Joystick=Movement Button=Magic stars

COMMODORE VIC-20 PRICE LIST

Effective October 1981

Order Number	Product Name	Retail Price
HARDWARE		
VIC20	**VIC 20 — The Friendly Computer** Commodore's revolutionary personal computer features color, sound, graphics, programmable function keys, built-in BASIC, expandable memory, low-priced peripherals and more! Connects to any TV or monitor. Includes RF modulator, switchbox, cables and self-teaching instruction book.	$299.95
VIC1530	**Commodore Datassette** Provides handy economical storage of user-written or pre-recorded programs using ordinary audio tape cassettes. Works like standard tape recorder, includes tape counter.	75.00
VIC1540	**VIC 1540 Single Disk Drive** Fast, high capacity storage and retrieval of data on standard 5¼-inch floppy diskettes. Stores up to 170K on each diskette, with read/write compatibility with PET/CBM computer systems. (December)	599.00
VIC1515	**VIC Graphic Printer** Economical dot matrix printer makes paper copies of BASIC programs, letters, business data and graphic displays. Connects directly to the VIC, prints all characters including letters, numbers and graphics. Prints 80 columns wide, 30 characters per second. Reliable tractor feed mechanism, Device 4/5 and test switch. Accepts sprocketed 8-inch roll or sheet paper.	395.00
SPECIAL CARTRIDGES		
VIC1210	**VIC 3K Memory Expander Cartridge** Plugs directly into the VIC's expansion port, expands memory to 8K RAM total.	39.95
VIC1110	**VIC 8K Memory Expander Cartridge** 8K RAM expansion cartridge plugs directly into the VIC. (October)	59.95
VIC1011A	**RS232C Terminal Interface** Provides interface between the VIC 20 and RS232 telecommunications modems. Connects to the VIC's **user port.** (October)	49.95
VIC1112	**VIC IEEE-488 Interface Cartridge** Provides interface between the VIC 20 and IEEE-488 instruments, including PET/CBM IEEE peripherals. Connects to the VIC's **expansion port.** (December)	99.95
APPLICATIONS SOFTWARE ON CARTRIDGE		
VIC1211A	**VIC 20 Super Expander** Everything Commodore could pack into one cartridge — 3K RAM memory expansion, high resolution graphics plotting, color, paint and sound commands. Graphic, text, multicolor and music modes. 1024 x 1024 dot screen plotting. All commands may be typed as new BASIC commands or accessed by hitting one of the VIC's special function keys. Includes tutorial instruction book. Excellent for all programming levels. (December)	69.95
VIC1212	**Programmers Aid Cartridge** More than 20 new BASIC commands help new and experienced programmers renumber, trace and edit BASIC programs. Trace any program line-by-line as it executes, pause to edit. Special KEY command lets programmers redefine function keys as BASIC commands, subroutines or new commands. (November)	59.95
VIC1213	**VICMON Machine Language Monitor** Helps machine code programmers write fast, efficient 6502 assembly language programs. Includes one line assembler/disassembler. (November)	59.95
RECREATIONAL GAMES ON CARTRIDGE		
VIC1901	**VIC AVENGERS** It's an invasion of space intruders and you're the VIC "Avenger." Space action for arcade enthusiasts. (November)	29.95
VIC1904	**SUPERSLOT** Colorful slot machine game works just like the real thing! Great music and sound effects! (October)	29.95
VIC1906	**VIC SUPER ALIEN** You're trapped in a maze and your only defense is the "alien buster". Can you capture the aliens before they zap you? (October)	29.95
VIC1907	**SUPER LANDER** Pilot your "Jupiter Lander" through the treacherous crevices of a mysterious planet. Variable rocket thrust, anti-gravity, horizontal retros. (October)	29.95

Order Number	Product Name	Retail Price
VIC1908	**DRAW POKER** Casino-style poker recreates the real thing! Superb animation and sound effects add to the fun, mystery and luck. (October)	29.95
VIC1909	**MIDNIGHT DRIVE** Authentic night driving simulation provides thrills, chills and . . . spills? An unusual computer challenge.	29.95

COMPUTER PROGRAMS ON TAPE

The following prerecorded programs are designed for use with the Commodore Datassette Tape Recorder. Programs on tape come in several varieties and are color coded by category as follows: Recreation (red), Education (blue), Business/Calculation (green), Home Utility (orange) and Computing Aids (black).

VT 106 A	**Recreation Program Pack A*** Car Chase — Fast-paced road action VIC 21 — Casino-style blackjack Blue Meanies From Outer Space — Space game Biorhythm/Compatibility — Compare biorhythms Spacemath — Math improvement grades 1-6 Slither/Super Slither — dexterity game	59.95
VT 107 A	**Home Calculation Program Pack A** Personal Finance I — Home budget Personal Finance II — Home budget VIC Typewriter — Word processor for home use Expense Calendar — Income, expenses, appointments Loan & Mortgage Calculator — Decision-making aid Home Inventory — Home belongings list. (October)	59.95
VT 164	**Programmable Character Set/Gamegraphics Editor** Lets the VIC user create up to 64 programmable characters and use them in BASIC programs. The Editor takes only one-half kilobyte of program space, works with tape, disk and printer. (October)	14.95
VT 232	**VICTerm I — Terminal Emulator** A handy VIC terminal program on tape which converts any VIC to a terminal for use with a telephone modem. (See RS232 Interface) (October)	9.95

TEACH YOURSELF PROGRAMMING SERIES

This 5-part series of tapes and manuals provides a self-teaching course in computer programming, from introductory BASIC to 6502 Machine Language Programming.

VL 101	**Introduction to Computing** Sample programs in book and on tape make this an interesting introduction for the non-computerist. Stresses "computing", not "programming." (October)	24.95
VL 102	**Introduction to BASIC Programming** A gentle but thorough introduction to BASIC programming. Excellent first book for any new computerist. Tutorial lesson tapes included. (November)	24.95

VIC 20 BOOKS AND MANUALS

VM 101	**PERSONAL COMPUTING ON THE VIC 20** The "friendliest" computer instruction guide available. This owner's manual comes free with every VIC 20 but is also in demand by teachers who use it in the classroom, and by "VIC families" who want more than one guide for each family member.	5.95
VM 110	**VIC 20 PROGRAMMERS REFERENCE GUIDE** The master VIC 20 reference manual includes information on VIC BASIC, 6502 Machine Code Programming, Input/Output ports, VIC microprocessing chips, and tips for all levels of programmers. Indispensable. (November)	16.95

ACCESSORIES

VIC 1515P	**VIC GRAPHIC PRINTER PAPER** 1000 sheet pack, tractor feed, 15 lb. bond	15.00
VIC 1515R	**VIC PRINTER RIBBON CARTRIDGE** for VIC Graphic Printer	9.95

Perfectly Balanced

educational software from MICRO-ED for PET® and VIC®

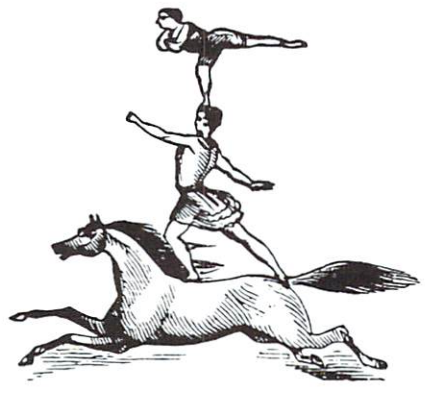
Send for our free catalog*

*please specify PET or VIC

MICRO-ED, Inc. • P.O. Box 24156

Minneapolis, MN 55424

or telephone us at (612) 926-2292



VIC and PET are the registered trademarks for Commodore Business Machines.

Machine Language Programming: Step 2

In the last issue, you were introduced to Machine Language programming via a short example typed in through the Machine Language Monitor. We also tried to give you some understanding of how the PET memory operates and by now you may have gotten yourself into trouble by Peeking and Poking around. In this article, we will take a closer look at that program, and introduce you to Registers, Addressing Modes, Condition Codes, and the 6502 Instruction Set.

But Where To Start??

Basically machine language programming is very similar to higher level language programming (BASIC), but the programmer must pay very strict attention to detail. What are the advantages? For one thing, a program written in machine language runs hundreds of times faster than a comparable program written in BASIC. This is because a BASIC program must be interpreted into machine language as the program is running. When a program is written in machine language, on the other hand, this intermediate step is not necessary. Another big advantage is the fact that programming in machine language allows you to do many things you can't do in BASIC. Wouldn't it be nice to be able to write a BASIC program that displays a data entry mask to the screen without having to manipulate the cursor each time you want to print. For example:

```
    10 pring @(10,30);''name: ''
```

This could mean to print ''name: '' at column 10, row 30 on the screen. Obviously this statement is invalid in the BASIC we know, but we could write a machine language program that allows us to do this from BASIC.

As was said in the previous article, machine language keyed in using the Machine Language Monitor is coded in Hexadecimal. This means that we will be using actual machine code instructions, such as ''a2'', rather than Assembler mnemonics, such as ''LDX.'' These codes are listed on the MCS6500 Instruction Set Summary card (included in the last issue), along with the associated Assembler mnemonics.

Registers

What's a register? Well, you won't find this definition in your Funk & Wagnall's. A register(s), in computer jargon, is a very specialized storage location(s) that is built into the CPU (the 6502 is the CPU in our case). Why do we need registers? There is a need for a place to store information, such as a variable, while the CPU is manipulating the data contained in that variable.

The 6502 microprocessor provides us with three registers to store data. The gotcha! here is that the highest number we are allowed to store in any of these areas is decimal 255 or hex ff. These three registers are called the Accumulator, the ''X'' Register and the ''Y'' Register. These registers are basic to the whole 6502 instructions set because almost every instruction needs then to perform its particular function.

The Accumulator

All operations effecting memory must be communicated through the Accumulator or the ''X'' or ''Y'' registers. The Acumulator is used as temporary storage in moving data from one memory location to another, and also as an interim storage area for a series of operations whereby the result of the operations is stored here. This means that the Accumulator is a special register because it is the only register that can be used for adding, subtracting, and other major functions performed by the 6502.

The ''X'' and ''Y'' Registers

These registers are very similar to the Accumulator although their primary purpose is as a storage area when moving data from one memory location to another. They are also sometimes used as counters.

The Condition Codes

Another register we will eventually become familiar with is called the Status Register. This register is the storage location for what are called the Condition Codes. It is possible to have eight Condition Codes represented, although there are only six that we can access through machine language. Of the other two possible codes, one was left open for expansion, and the other can only be accessed by the microprocessor. How can eight codes be stored in one location? The answer to this question lies in the explanation of an 8-bit register.

As was mentioned above, the highest number that can be represented in one byte is 255. This is because in the binary number system, a number is represented in 1's and 0's (bits), and one byte can contain a maximum of eight bits. The highest binary number that can be stored in one byte is 11111111, which in decimal is 255. You should start to think at this point in 8-bit notation, which just means that any number between 0-255, is thought of and represented using all 8 bits. For example, even ''0'' is represented as ''00000000.''

In the status register, we assign each of these bit positions a number from 0 through 7 (to confuse us of course, the

➡

Corrections!!

I have discovered some errors in the last article (Programming in Machine Language—October, 1981) which I wish to bring to our readers' attention. The first was on page 64. In the middle column, second paragraph, I was discussing how much RAM you have available when you power up the system, and mentioned that there was a discrepancy of one byte between what the power on message displayed and what we calculated. The real reason for the discrepancy is that the first byte of Text RAM (location 1024) is always initialized to zero (0) and so your BASIC program always starts at location 1025.

The second error I discovered was on the sample program listed on page 66. Although the program as listed, would run with no problem, one instruction was left out. Also three extra bytes at the end were not necessary. The program should actually have been listed as follows:

```
    .: 0408  a2  0d  8a  20  d2  ff  a2  20
    .: 0410  8a  20  d2  ff  e8  e0  60  d0
    .: 0418  f7  00 <RETURN>
```

—Dave Scott

eighth bit is really bit "7"), starting from the right hand side. So each condition code is actually bit number "1". The figure below will help explain this.

```
7  6  5  4  3  2  1  0 ←—— Bit position
N  V     B  D  I  Z  C ←—— Condition Code
0  0  0  0  0  0  1  0 ←—— Sample number
```

In the example above, the number stored in the status register is decimal 2, which in binary is 10, or in 8-bit notation, 00000010. As you can see from this, when the number 2 is stored, only bit "1," the "Z" bit, has a 1 in it. We generally say in this case that the "Z" bit is "set."

So what is a Condition Code and what is it used for? As you can see from the explanation above, any bit can be on (1) or off (0). It is this concept of a switch which explains the function of a Condition Code. A code (bit) is set on if the "condition" it represents occurs as a result of the previous instruction. For instance, if we had added a number to the Accumulator and the result of that operation was "zero," then the "Zero" Condition Code (the "Z" bit) would be set to "1."

Because of this important feature, the 6502 instruction set contains certain instructions which can test the status of these codes and perform the intended operation if the switch is set on (1), or bypass the operation if the switch is off(0). This will become more clear in a moment.

If you are totally confused by now, don't feel bad, it gets worse. But don't worry, registers, condition codes, etc., all become understandable when you begin programming in machine code.

Addressing Modes

When any machine language instruction is executed, there are a number of ways in which data in memory can be "addressed." What does that mean? Well, basically, data that you want to act on is stored at some particular "address" in memory or, in the simplest case, is given as part of the instruction, or constant. In this latter case you actually program, as part of the instruction, the value you want acted upon in your program. For instance, you may want to check if the value stored in the "X" register is equal to "2," which is the constant value. This type of addressing is called "Immediate," because the given value is expressed "immediately" following the instruction.

In the cases where the data to be acted upon actually resides in memory, there are a number of different addressing modes available. Two other modes that we will be concerned with in this issue are "Relative" and "Implied." In 6502 machine language there are certain instructions which allow you to transfer control fom one part of the program to another (analagous to, but not the same as, the GOTO in BASIC. These instructions use what is called "Relative" addressing. The term "relative" comes from the fact that when one of these "branch to" instructions is executed, the value following the instruction specifies how many bytes forward or backward from the current instruction (using the program counter) the program should skip before continuing execution. In other words, this value represents a location in memory which is relative to the position of the "branch to" instruction.

In the "implied" mode, the location of the data to be acted upon is actually a part of the microprocessor, and as such, is "implied" by the nature of the instruction. These storage locations, such as the Accumulator, the X and Y registers, and other registers (which we will talk about later), are built in to the microprocessor. So for example, when we want to decrease the value currently stored in the Y register by one (commonly known as "decrementing"), the microprocessor supplies us with an instruction to do exactly that. The instruction "implies" that it is the Y register within the microprocessor that we wish to act on.

The final mode of addressing we will discuss here is the "absolute" mode. In this mode, the memory address that is given as part of the instruction, is the exact, or absolute location of the data to be acted on.

There are a number of other addressing modes which you must become familiar with in order to do any advanced machine language programming, but we will discuss these later.

The 6502 Instruction Set

The best way to gain an understanding of machine language programming and the 6502 instruction set is to look at and write some examples. Let's take a look at the sample program from last issue (October, 1981) in more detail. The code for the program from last issue is as follows:

```
001     a2 0d       ldx #0d
002     8a          txa
003     20 d2 ff    jsr ffd2
004     a2 20       ldx #20
005     8a          txa
006     20 d2 ff    jsr ffd2
007     e8          inx
008     e0 60       cpx #60
009     d0 f7       bne f7
010     00          brk
```

As you can see, we have grouped the program into the appropriate instruction segments. We have also listed the associated Assembly language mnemonics in order to better illustrate the function of each instruction. In this listing, the instruction is the first hex number in the leftmost column.

Line 1—"a2 0d"

The first line of the program introduces us to the a2, Load the X register (immediate addressing), instruction. The a2 instruction is one of 5 different forms of the instruction to load the X register with data. The a2 form of "LDX" (the Assembler mnemonic for load the X register) is the "immediate" form of the instruction. In other words, the data to be loaded into the register immediately folows the instruction. In this case, the value "0d" follows the instruction, so the first instruction loads 0d (0d is hex for decimal 13, which is the ASCII code for a carriage return) into the X register. To indicate an immediate hex value in Assembler, the pound sign ("#") precedes the value.

Line 2—"8a"

This line shows us the format of the 8a, Transfer data from the X register to the Accumulator, instruction. This instruction, quite simply, puts a copy of the data that currently exists in the X register into the Accumulator. The data in the X register remains unchanged. There is only one form, the "implied" mode, for this instruction. In our example then, the value "0d" is copied to the Accumulator.

Line 3—"20 d2 ff"
The 20, or JSR (Jump to subroutine) instruction, will make the program branch to a new location. It functions exactly like the GOSUB in BASIC so that when a RETURN is executed program execution is transferred back to the point after the instruction. Its only form is "absolute" because the value that follows it is the exact or absolute location to which the program will branch. You will notice that this instruction requires three bytes, one for the instruction, and two for the branch-to memory address. In this case, the memory address the program will jump to is "ffd2." We read this as "ffd2" rather than "d2ff," because this instruction, along with some other instructions which act on memory locations, will take the first byte (d2) as the "low" byte of the address, and the next byte as the high byte of the address. This method is built-in to the 6502 and therefore, must always be programmed this way.

Why did we use the address "ffd2"? Because this is a location in ROM (it comes with the machine) that already contains a subprogram which will output a character to the screen. We can use this and other subprograms built into ROM provided we perform the necessary setup operations. In this case, we need only to load the character we wish to print into the Accumulator. Once it is there, we can jump to this subroutine, and the character will be printed on the screen. If you look at the Memory Map provided in the last issue, you will see that address ffd2 is the starting address of the routine that outputs a character.

So the first three instructions or lines of code are used to print a carriage return. You may be wondering why we first loaded the X register with a value, only to transfer it to the Accumulator. We could have loaded the Accumulator directly, but the data in the Accumulator gets manipulated or altered by the same instructions. This method assures that you have a copy of the original data.

Line 4—"a2 20"
The fourth line is another example of loading the X register with a value. In this case we are loading the "immediate" hex value "20." This value is the numerical representation of a space character. (In fact, all characters that can be printed are assigned a numerical value between the number 0 and 255.) These values are usually standard from machine to machine, and as such an industry standard for this coding scheme has developed—ASCII codes. Because of the special graphics and reverse video characters available on the PET, this coding scheme is not always matched correctly. In any case, the value "20" hex represents the first printable character in the PET character set.

Line 5—"8a"
Again we will transfer the value in the X register to the Accumulator, to prepare for the subroutine that prints a character. We do not, in this case, want to load the Accumulator directly because we will need to manipulate the character in the X register further on in the program.

Line 6—"20 d2 ff"
Next, jump to the subroutine that prints the data that is in the Accumulator, namely a space (hex 20).

Line 7—"e8"
The e8, or INX instruction will increment the value that is currently in the X register by one and overwrite the current X register data with the new value. As we mentioned above, the addressing mode is "implied" because the nature of the instruction implies manipulation of the X register. After this instruction is performed, the X register will contain a new value, hex 21. This is actually the next character in the PET character set that we will print, the character "!".

Line 8—"e0 60"
The e0, or CPX (Compare the data in the X register with memory) instruction compares the data in the X register with the immediate value following the instruction, in this case hex 60. This is the "immediate" form of the CPX instruction, thus "CPX #60". We are comparing the value stored in the X register with a hex 60 because this number is the numeric representation for the character following the last character we wish to print. When this instruction is evaluated, it will subtract the value stored in the X register from the immediate hex value 60. If the result is zero, i.e., the value in the X register equals 60, then the "Z" bit is turned on, meaning that when the operation was evaluated zero was the result. (The result of a compare is not stored anywhere.)

Line 9—"d0 07"
The d0, or BNE (Branch if the result of the previous operation is not equal to zero) instruction will test the state of the "Z" bit, and branch to the address specified by the instruction if this condition code is not turned on. In the previous line we saw that the CPX instruction would set the "Z" bit only if the value in the X register was a hex 60. So this instruction will branch until the data stored in the X register equals hex 60. When this condition occurs, the "Z" bit is on, so the branch is not taken, and program execution continues with the next instruction.

Since this is a "relative" addressing instruction, the branch-to address is relative to the instruction itself. The address specified is hex "f7" away from this instruction. Hex "f7" is really −9, so that the effective branch-to address is 9 bytes "back" from the current instruction. The catch here is that the current instruction is now at line 10, "00". This is because the instruction at line 9 has already been executed. So the program will branch back 9 bytes from line 10; this brings us back to line 5, which is the instruction "8a" (count to prove it).

Line 10— "00"
The 00, or BRK (Break) instruction will terminate program execution. In our example, this instruction will only be executed when the "Z" bit is set—the branch instruction will not be executed.

To summarize, the program first prints a carriage return (lines 1-3). Next it stores a starting value in the X register (line 4). Then a loop is entered whereby the first instruction executed each time the loop is restarted is to transfer the value in the X register to the Accumulator (line 5). Next it prints out that character (line 6), adds one to the value in the X register which gives us the next character to print (line 7), tests to see if the new X register value is equal to 60 (line 8), branches to the beginning of the loop if it's not (line 9), and finally falls out of the loop and stops the program when the value is equal to 60. Easy, right?? Try changing the byte that sets the first value to be printed (line 4) and the byte that checks for the last value to be printed (line 8). Try the values "a0" and "ff" respectively, and see what happens.

Next time, we'll look at another example, and more instructions.■

—Dave Scott & Patrick McAllister

On PEEKing the PET Keyboard

by
Elizabeth Deal
Malvern, PA

In the October issue there appeared an article of mine called "Four Pet Keyboards." I would like to make an important amendment to it.

The emphasis of the article was on simple conversion between existing keyboards. When used as such, the charts can help convert a program that already contains PEEK(151) type of keyboard check between various models of the PET/CBM system. This part remains valid and can be so used.

However the article also suggested, or at least implied, a benefit in using PEEK(151) in your programs. Some other work I was doing caused Jim Butterfield to react to that use. He discourages use of PEEK(151) primarily because of the very necessity of converting, as any need for conversion makes programs less transportable between machines.

He has no problem with using it to check if a key was pressed as in 10 IF PEEK(151=255 GOTO10, but no more.

It's a sensible approach and I therefore concur with the policy and retract that part of my article suggesting such use. If you already have programs containing PEEK(151), then by all means use the conversion charts. Otherwise, it seems better to stay away from checking 151 for particular key numbers. There are better ways of doing the job without creating incompatibility problems.

And while we're on a subject of corrections, readers of July '81 issue should substitute S2 (letter S as in SAM, followed by number two) for 52 in a P2= . . . expression on page 40, point 8 and should insert a missing (but not that important) line 250 GOSUB 490 in a program on page 42. Sorry.■

Bits and Pieces

Reprinted from
the TRANSACTOR

Unit to Unit Copying

Ever needed to copy one or two programs from 4040 to 8050 (or Vice Versa)? So you pull out your Change Unit Address program, then Copy All, and after answering all the prompts, the files you've selected get transferred. Seems like a lot of work for only a couple of files, doesn't it.

Alas, there is another way! Connect your 4040 and 8050 to the PET/CBM and leave the device numbers alone (i.e. both device 8). Assuming we're going from 4040 to 8050, insert your source disk in drive 0 of the 4040 and place the destination disk in drive 1 of the 8050. Now DLOAD the program from 4040 drive 0. Both disk units will fire up, but the 8050 will give an error since drive 0 is empty . . . So What! DSAVE to drive 1 and the 8050 error light will go off and the 4040 error light comes on. Again . . . So What! Check the directory and you'll find the transfer took place without a hitch!

This would also work for SEQ files with a small bit of software that knows how to ignore anticipated errors. Of course, for a lot of files or programs, the Copy All approach would probably be quicker, but isn't it nice to know you can deliberately cause disk errors and still accomplish something!

Disabling The STOP KEY

Here are a couple of ways to disable the 4.0 STOP key. If you don't need the internal clock:

Disable with	POKE 144, 88
Restore with	POKE 144, 85

Notice that to disable you simply add three to low order address of the IRQ vector, as with all other versions of Commodore Basic.

If you do need the clock, Jim Butterfield has a tidy little Basic routine that does it:

```
100 D$ = "20>:??:9??8=9;004<58>4"
110 FOR J = 1 TO LEN (D$)/2
120 PIKE J+852, ASC(MID$(D$,J*2-1))*16
    + ASC(MID$(D$,J*2))-816
130 NEXT
```

To engage the routine: POKE 145,3
To reset the rector: POKE 145,228■

Programs to WordPro Conversion

This tidy little Basic program converts programs to WordPro files.

The program does not convert special characters such as cursor graphics. These would have to be changed to CHR$(values or some other representation which would be unfeasible. Instead, these characters are left alone and show up in WordPro just as they do in a normal LIST. From here, you can use WordPro to edit them to suit (and what better editor). The program was actually used here to convert itself for publication. Then alpha characters were substituted for cursor characters that don't print on the NEC Spinwriter. (remember, ''left-arrows'' can't be printed from Word Pro)

```
100 dim a$(90):for i=0 to 90:read a$(i):next
110 print ''program filename '';:gosub 890:fi$=a$
112 print ''wordpro filename '';:gosub 890:wf$=a$
115 input ''device number       8[CR CR CR]'';dv
120 open 2,dv,2,fi$+'',p'':gosub 880:get#2,a$,a$
121 open 3,dv,3,wf$+'',p,w'':gosub 880:print#3,
    chr$(192)chr$(91);
125 sl=0:get#2,a$,a$:if a$=::'' then 600:rem skip link
    unless end of program
126 print ''[HM HM CLR]''chr$(14);:get#2,a$,B$:rem
    get line number
140 n=asc(a$chr$(0))+asc(b$+chr$(0))*256:print n;
150 get#2,a$:p=asc(a$+chr$(0)):if p=0 then print
    ''[left-arrow]'':goto500
160 if (peek(205)<>0) or (p<128) then print chr$(p);:
    goto 200
170 printa$((p-128);
200 if (a$='':'' or a$='','') and (peek(198)>45) then
    220
201 if peek(198)>55 then 220
210 goto150
220 print ''[left-arrow]'':print n;chr$(150);:sl=
    sl+1:goto 150
500 for 1=0 to sl:q=32768+1*80
505 for i=0 to 79:p=peek(q+i):print#3,chr$(p);: next
510 next
520 goto 125
600 close 3:close 2:end
880 if ds<20 then return
885 print ds$:end
890 poke 623,34:poke 624,27:poke 158,2:input a$: return
900 data end,for,next,data,input#, input,dim,read,
    let,goto,run,if,restore,gosub
910 data return,rem,stop,on,wait,load,save,verify,
    def,poke,print#,print,cont
920 data list,clr,cmd,sys,open,close,get,new,
    tab(,to,fn,spc(,then,not,step,+,-
930 data *,/,  ,and,or,>,sgn,=,<,int,abs,usr,fre,←
    pos,sqr,rnd,log,exp,cos,sin
940 data tan,atn,peek,len,str$,val,asc,chr$,left$,
    right$,mid$,go,concat
950 data dopen,dclose,record,header,collect,backup,
    copy,append,dsave,dload
960 data catalog,rename,scratch,directory ■
```

—Paul Higginbottom

Video-Typewriter and Typewriter 4

by
Preston F. Marshall

Many personal computer owners have a limited use for some kind of simple word processing but do not have sufficient motive to buy the memory, disks, etc., that makes the full package and the full price tag. One solution is some form of "Video-Typewriter." As an example, an 8K PET with a tape recorder and an added $500 AXIOM printer has worked quite well as a video-typewriter.

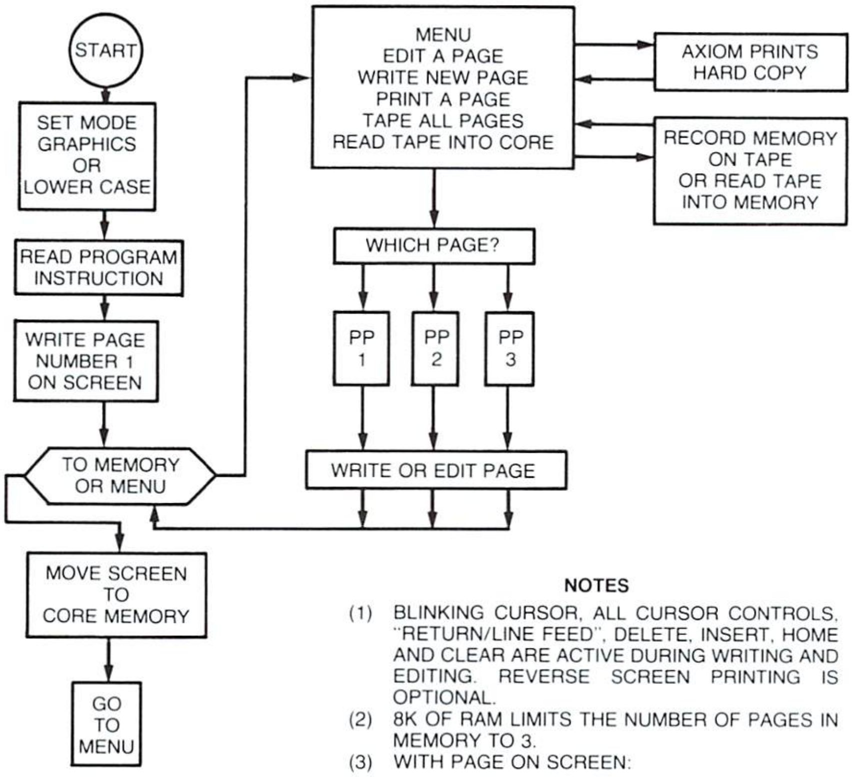
The advantage of being able to use the screen editing functions makes it all worthwhile and partially negates poor typing. Copy from the AXIOM can be photocopied to eliminate the disadvantages of the coated paper. For more formal documents, weekend drafts are taken to the office and retyped.

Although the program listed is specifically designed for the PET, the flow chart for the program can be followed to produce the same results with any other micro.

For those who have the incentive to pay for a true word processing system, the video-typewriter approach is inadequate. The time taken to put a page into memory is excessive, and a magnetic cassette storage is much too slow. If, however, your needs and budget are limited, it can come down to a selection of the video-typewriter or nothing.

For those who would opt for a faster reading of a page into memory and a much slower printout, a second program, (typewriter 4) is listed that accomplishes this aim. In this case, the translation of PET code is done during printing, and the pages are poked in PET code into an area of memory that is reserved by moving the "top of memory" pointer down. In the first program (Video-Typewriter), code conversion is made during storage of a page into memory as strings. As more strings go into memory, old string material is shifted from one memory location to another, causing a progressively longer storage time as more pages are added.

"PET VIDEO-TYPEWRITER"



NOTES

(1) BLINKING CURSOR, ALL CURSOR CONTROLS, "RETURN/LINE FEED", DELETE, INSERT, HOME AND CLEAR ARE ACTIVE DURING WRITING AND EDITING. REVERSE SCREEN PRINTING IS OPTIONAL.
(2) 8K OF RAM LIMITS THE NUMBER OF PAGES IN MEMORY TO 3.
(3) WITH PAGE ON SCREEN:

[→] SIDE ARROW IS GO TO MENU.

[↑] UP ARROW IS SCREEN TO MEMORY.

(4) USES OLD OR NEW ROM's.

REQUIRES

(1) PET
(2) AT LEAST 8K OF RAM
(3) AXIOM PRINTER
(4) TAPE RECORDER

Both programs follow the same flow chart and produce the same results.

Since the PET has different ROM sets with different sets of memory locations, both programs were made to work on either set of ROMs.

For those using a printer other than the AXIOM, the "OPEN" statement for the printout may have to be modified for that specific printer.

In my opinion, the increased utility these programs give your computer is well worth the time taken to key them in.

```
0:v*




 100 rem********************************
 200 rem this is vidio- typewriter
 210 rem    by pres marshall 2/8/81
 220 rem********************************
 1020 pp=1:rv=0:rem page number-not reverse screen
 1025 data 64,0,128,64,-64,-128,0,-64:rem peek to asc conversion
 1030 dim a$(71),da(7)
 1035 for i=0 to 7:read da(i):next i
 1040 kb=525:ce=548:cc=549:cf=551:rem old rom's
 1045 if peek(51234) then ce=167:cc=168:cf=170:kb=158:rem new rom's
 1050 poke 59468,14:rem lower case
 1060 for i=0 to 71:a$(i)="":next i
 1100 print "▓▓  this is program vidio-typewriter           by pres marshall
 1110 print:print:print "uP TO 3 SCREEN LOADS CAN BE WRITTEN,"
 1120 print "EDITED AND THEN TYPED."
 1130 print:print:print "do you wish to use graphic mode or lower case mode?"
 1140 print:print:print "▓    press +g| or +l|    ▓
 1150 va$="gl"
 1160 gosub 61000:rem do a checked get
 1170 if c$="g"then sm=12:am=15:rem screen mode and axiom mode set to graphics
 1180 if c$="l" then sm=14:am=14
 1190 poke 59468,sm
 1200 rem program instructions begin here
 1210 print"▓you can now hit any keyboard key to "
 1220 print"enter the program and start writing"
 1230 print"copy. use all the cursor and edit keys."
 1240 print:print:print"when you are finished with a page press"
 1250 print"the up arrow +▒| not the up cursor."
 1260 print:print:print"there will be a delay while the page is read into core m
emory."
 1262 print "reading into memory takes 60 seconds"
 1265 print:print
 1270 print"to avoid reading the page into core,"
 1280 print"press the side arrow +▒|. this method "
 1290 print"of avoiding the read is used when"
 1300 print"checking over the copy.
 1400 poke kb,0:wait kb,1:geta$:a$=""
 3000 rem write to screen or edit screen
 3010 print"▓";
 3020 print"▓";
 3040 poke ce,0:rem enable cursor
 3050 if peek(kb)=0 then 3050
 3055 get a$
 3057 poke cc,5
 3060 if peek(cf)=1 then 3060
 3070 if a$="▒" then goto 4000
 3080 if a$="▒" then goto 5000
 3090 print a$;:goto 3050
 4000 rem read screen into array a$(x)
 4005 for i=24*(pp-1) to 24*(pp-1)+23:a$(i)="":next i
 4010 for i=24*(pp-1) to 24*(pp-1)+23
 4015 pz= 32768+40*(i-24*(pp-1))
 4020 for k=0 to 39
 4022 pk=peek(pz+k)
 4024 if pk/128>=1 and rv=0 then r$=chr$(18):rv=1:goto 4030
 4028 if pk/128<1 and rv=1 then r$=chr$(146):rv=0:goto 4030
 4029 r$=""
```

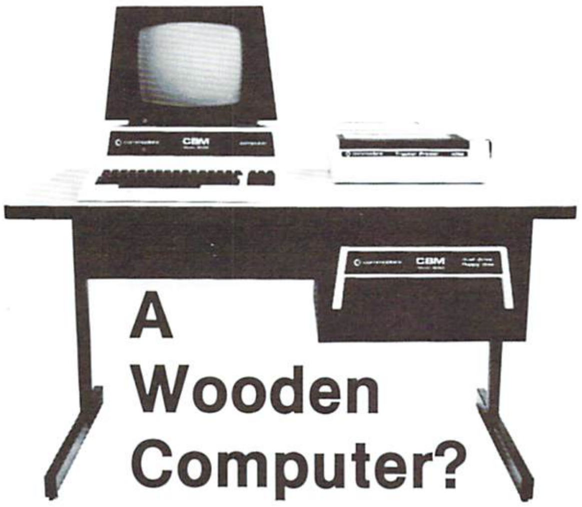
→

```
 4030 a$(i)=a$(i)+r$+chr$(pk+da(pk/32))
 4050 next k,i
 5000 rem menu starts here
 5010 print "S        you just left page";pp
 5020 print:print:print" do you wish to:"
 5030 print:print"                           press"
 5040 print"print on axiom          p"
 5050 print"edit another page       e"
 5060 print"write a new page        w"
 5070 print"tape all pages          t"
 5080 print"read pages from tape    r"
 5081 print:print"if you crash press stop and goto 5000
 5082 va$="pewtr"
 5084 gosub 61000:rem do a checked get
 5090 print:print:print"you pressed ";c$
 5095 if c$="p" then goto 51000
 5096 if c$="t" then goto 62000
 5097 if c$="r" then goto 63000
 5100 print"now enter the new page number and check  it before hitting return."
 5110 input tt:rem temporary test number
 5120 if tt<1 or tt>3 then print"wrong entry enter 1,2 or 3":goto 5110
 5130 pp=tt
 5160 if c$="e" then goto 50000
 5170 if c$="w" then goto 3000
 50000 rem prints out old page on screen  5190
 50006 print "S";
 50010 for i=24*(pp-1) to 24*(pp-1)+23
 50020 print a$(i);
 50030 next i
 50040 print "page number ";pp;
 50050 goto 3020
 51000 rem state page numbers for print out
 51010 print "you want to print hard copy.be sure the  printer is on line."
 51030 input "input starting line-0 to 71. pp1=0 pp2= 24 pp3=48";sl
 51040 if sl<0 or sl>71 then print "try aagin-remember 0 to 71":goto 51010
 51050 input"input ending line 0 to 71 pp1=23 pp2=47  pp3=71";el
 51060 if el<0 or el>71 then print"try again-remember 0 to 71":goto 51010
 51070 if sl>el then print"your starting line was > than your last":goto 51010
 51080 goto 60000
 60000 rem axiom print out begins here
 60010 open 4,4:cmd4:printchr$(8)+chr$(am):print#4,
 60020 for i=sl to el
 60030 print#4,a$(i)
 60040 next i
 60050 close 4:goto 5000
 61000 rem sub to validate the response to any get
 61010 get c$:if c$="" then 61010
 61020 for x=1 to len(va$)
 61030 if c$=mid$(va$,x,1) then return
 61040 next x:goto 61010
 62000 rem start of record  a$(x) from   core to tape
 62010 poke 243,122:poke 244,2:open 1,1,1
 62020 for i=0 to 71:print#1,chr$(34)+a$(i)+chr$(34):next i
 62030 close 1:goto 5000
 63000 rem start of read a$(x) from tape
 63010 poke 59411,53:open 1,1,0
 63020 for i=0 to 71:input#1,a$(i):next i
 63030 poke 59411,61:close 1:goto 5000
ready.
ready.

 100 rem********************************
 200 rem this is vidio- typewriter 4
 210 rem    by pres marshall 2/20/81
 220 rem********************************
 1020 pp=1:rv=0:rem page number-not reverse screen
 1025 data 64,0,128,64,-64,-128,0,-64:rem peek to asc conversion
 1030 dim da(7)
 1035 for i=0 to 7:read da(i):next i
 1040 kb=525:ce=548:cc=549:cf=551:lo=134:ho=135:rem old rom's
 1045 if peek(51234) then ce=167:cc=168:cf=170:kb=158:lo=50:ho=51:rem new rom's
 1050 poke 59468,14:rem lower case
 1052 poke lo,190:poke ho,20:rem poke top of memory down to 5310 to make room
 1053 rem for 3 pages of screen above 5310 at 960 bytes per page of 24 lines
 1100 print "▒▒  this is program vidio-typewriter         by pres marshall ▒"
 1110 print:print:print "uP TO 3 SCREEN LOADS CAN BE WRITTEN,"
 1120 print "EDITED AND THEN TYPED."
 1130 print:print:print "do you wish to use graphic mode or lower case mode?"
 1140 print:print:print "▒    press +g| or +l|   ▒
 1150 va$="gl"
 1160 gosub 61000:rem do a checked get
 1170 if c$="g"then sm=12:am=15:rem screen mode and axiom mode set to graphics
 1180 if c$="l" then sm=14:am=14
 1190 poke 59468,sm
 1200 rem program instructions begin here
 1210 print"▒you can now hit any keyboard key to "
 1220 print"enter the program and start writing"
 1230 print"copy. use all the cursor and edit keys."
 1240 print:print:print"when you are finished with a page press"
 1250 print"the up arrow +▒| not the up cursor."
 1260 print:print:print"there will be a delay while the page is read into core m
emory."
 1262 print "reading into memory takes 25 seconds"
 1265 print:print
 1270 print"to avoid reading the page into core,"
 1280 print"press the side arrow +▒|. this method "
 1290 print"of avoiding the read is used when"
 1300 print"checking over the copy.
 1400 poke kb,0:wait kb,1:geta$:a$=""
 3000 rem write to screen or edit screen
 3010 print"▒";
 3020 print"▒";
 3040 poke ce,0:rem enable cursor
 3050 if peek(kb)=0 then 3050
 3055 get a$
 3057 poke cc,5
 3060 if peek(cf)=1 then 3060
 3070 if a$="▒" then goto 4000
 3080 if a$="▒" then goto 5000
 3090 print a$;:goto 3050
 4000 rem read screen into memory
```

→

```
4040 for i=0 to 959
4050 pk=peek(32768+i)
4060 poke(5311+960*(pp-1)+i),pk
4070 next i
4080 goto 5000
5000 rem menu starts here
5010 print "S         you just left page";pp
5020 print:print:print" do you wish to:"
5030 print:print"                              ' press"
5040 print"print on axiom             p"
5050 print"edit another page          e"
5060 print"write a new page           w"
5070 print"tape all pages             t"
5080 print"read pages from tape       r"
5081 print:print"if you crash press stop and goto 5000
5082 va$="pewtr"
5084 gosub 61000:rem do a checked get
5090 print:print:print"you pressed ";c$
5095 if c$="p" then goto 51000
5096 if c$="t" then goto 62000
5097 if c$="r" then goto 63000
5100 print"now enter the new page number and check  it before hitting return."
5110 input tt:rem temporary test number
5120 if tt<1 or tt>3 then print"wrong entry enter 1,2 or 3":goto 5110
5130 pp=tt
5160 if c$="e" then goto 50000
5170 if c$="w" then goto 3000
50000 rem prints out old page on screen  5190
50010 print "S";
50020 for i=0 to 959
50030 pk=peek(5311+960*(pp-1)+i)
50040 poke(32768+i),pk
50050 next i
50060 print "SQQQQQQQQQQQQQQQQQQQQQQQQpage number=";pp;
50070 goto 3020
51000 rem state page numbers for print out
51010 print "you want to print hard copy.be sure the  printer is on line."
51030 input "input starting line-0 to 71. pp1=0 pp2= 24 pp3=48";sl
51040 if sl<0 or sl>71 then print "try aagin-remember 0 to 71":goto 51010
51050 input"input ending line 0 to 71 pp1=23 pp2=47  pp3=71";el
51060 if el<0 or el>71 then print"try again-remember 0 to 71":goto 51010
51070 if sl>el then print"your starting line was > than your  last":goto 51010
51080 goto 60000
60000 rem axiom print out begins here
60010 open 4,4:cmd4:printchr$(8)+chr$(am):print#4,
60020 for i=sl to el
60030 rv=0
60040 for k=0 to 39
60050 pk=peek(5311+40*i+k)
60060 if pk/128>1 and rv=0 then r$=chr$(18):rv=1:goto 60090
60070 if pk/128<1 and rv=1 then r$=chr$(146):rv=0:goto 60090
60080 r$=""
60090 print#4,r$+chr$(pk+da(pk/32));
60100 next k,i
60110 print#4,
60120 close4:goto 5000
61000 rem sub to validate the response to any get
61010 get c$:if c$="" then 61010
61020 for x=1 to len(va$)
61030 if c$=mid$(va$,x,1) then return
```

```
 61040 next x:goto 61010
 62000 rem start of record  from   core to tape
 62010 poke 243,122:poke 244,2:open 1,1,1
 62020 for i=5311 to 8190
 62030 pk=peek(i)
 62040 print#1,chr$(pk)
 62045 next i
 62050 print#1,
 62060 close 1:goto 5000
 63000 rem start of read from tape into core
 63010 poke 59411,53:open 1,1,0
 63020 for i=5311 to 8190:get#1,a$:get#1,b$  :pokei,(asc(a$)):next i
 63030 poke 59411,61:close 1:goto 5000
ready.
```



The specifications only confirm the obvious:

- •Cream and charcoal color beautifully matches the Commodore hardware and blends with your decor.
- •An ideal 710 mm (28") keyboard height yet no bumping knees because a clever cutout recesses the computer into the desk-top.
- •High pressure laminate on both sides of a solid core for lasting beauty and strength.
- •Electrostatically applied baked enamel finish on welded steel legs—no cheap lacquer job here.
- •T-molding and rounded corners make a handsome finish on a durable edge that won't chip.
- •Knocked down for safe, inexpensive shipment.
- •Patented slip joints for quick easy assembly.
- •Leveling glides for uneven floors.
- •Room enough for a Commodore printer on the desk, yet fits into nearly any den or office niche— H: 660 mm (26") W: 1170 mm (46") D: 660 mm (26").
- •Matching printer stand available with slot for bottom feeding.

Not from Commodore!

So why should the desk look like wood? A pleasant cream and charcoal trimmed desk looks so much better with Commodore systems. One look and you'll see. Interlink desks are right. By design.

PRICE: $299

In short, as Commodore dealers, we won't settle for anything that looks good only in the catalog! Our customers won't let us. They don't buy pictures. And neither should you. This is why **we will let you use one of our desks for a week** and then decide. If for any reason you don't like it, just return it in good condition for a cheerful refund.

If your Commodore dealer doesn't carry our desks yet, send a check for $299 and we will ship your desk freight paid!

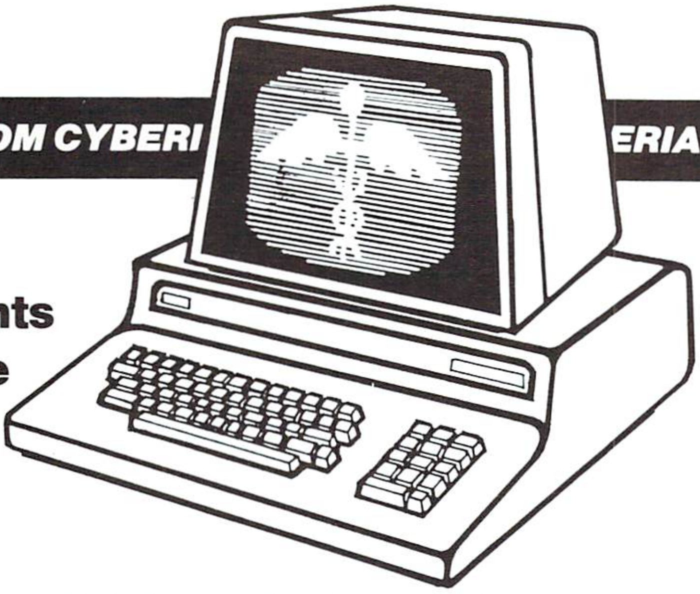
Name______________________________

Address____________________________

City________________ **St**_____ **Zip**______

Interlink, Inc., Box 134, Berrien Springs, MI 49103
Master Charge and Visa welcome. Call our order line:
616-473-3103

NEW FROM CYBERIA ★ NEW FROM CYBERIA ★ NEW FROM CYBERIA



Mari

A comprehensive accounts receivable and insurance billing system for modern health care offices and clinics

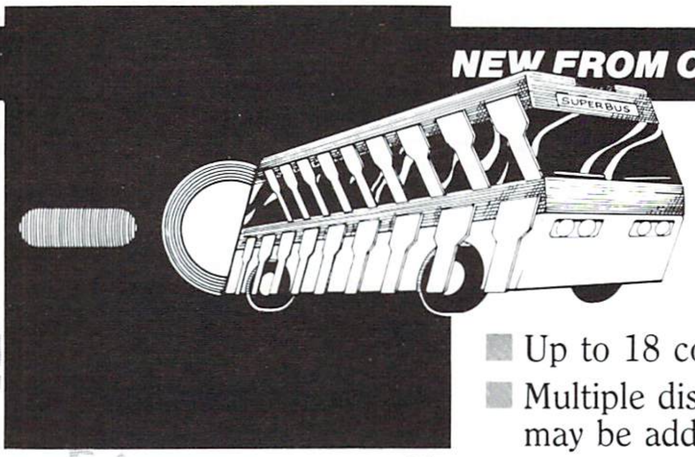
DESIGNED FOR COMMODORE 8000 SERIES COMPUTERS AND DISK DRIVES

- Help functions are always on-line
- Supports CPT, ICD and RVS medical coding
- Supports open item or balance forward accounting systems
- Interfaces to popular word processing programs
- Multiple terminals may be added with Superbus 4.0
- Multiple disk drives may be used—no limit on number of patients or accounts
- Includes a data base and forms generator to fill out any insurance form
- Includes a computer aided instruction program to train new users

Includes these standard reports:

- A/R aging
- A/R transactions
- General ledger
- Instant cash receipt
- Income analysis by physician
- Doctor referral report
- Patient/account cross reference
- Standard SuperBill insurance form

NEW FROM CYBERIA ★ NEW FROM CYBERIA



SUPERBUS 4.0

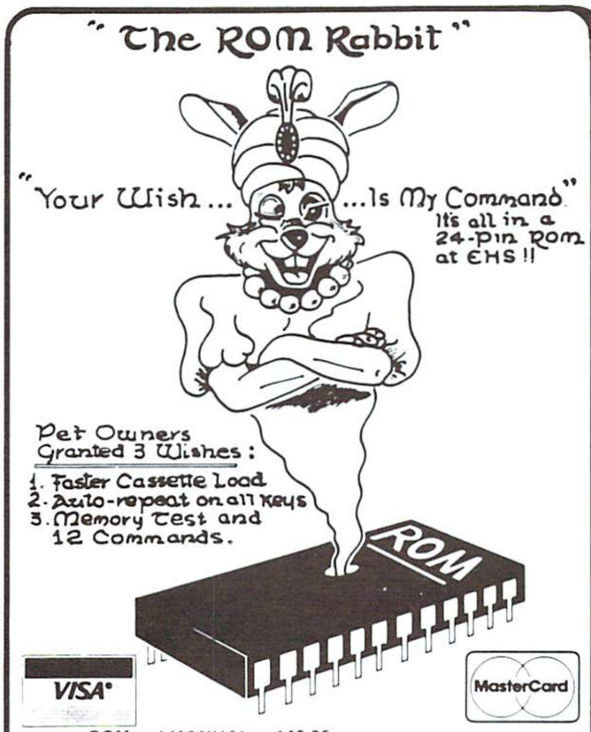
Commodore Computer Networking System

- Up to 18 computers can be interconnected
- Multiple disk drives, printers and other devices may be added to the network
- Sophisticated security system prevents unauthorized use
- WordPro and Wordcraft programs are supported to allow multiple terminal word processing

NEW ★ NEW ★ NEW

*WordPro is a trademark of Professional Software, Inc.

CYBERIA INC. 2330 LINCOLN WAY, AMES, IA 50010 • 515-292-7634


" The ROM Rabbit "
" Your Wish ... ...Is My Command"
It's all in a 24-pin ROM at EHS !!
Pet Owners Granted 3 Wishes :
1. Faster Cassette Load
2. Auto-repeat on all Keys
3. Memory Test and 12 Commands.
ROM
VISA
MasterCard

ROM and MANUAL — $49.95
Specify 3.0 or 4.0 Basic (Works with or without toolkit)

EASTERN HOUSE SOFTWARE
3239 Linda Drive
Winston-Salem, N. C. 27106

(919) 924-2889
(919) 748-8446

ROM OR ON CASSETTE

THE PET RABBIT

4.0 OR 3.0 ROMS

The PET Rabbit is a programmers aid which provides 12 additional commands that can be executed in BASIC's direct mode. In addition to the commands, automatic repeat of any key held down for 0.5 seconds is also provided. This will greatly aid inputtings of characters and provide more convenient cursor control. Most importantly, the RABBITs high speed recording technique allows an 8K program to be saved in 38 seconds instead of the normal 2 minutes and 44 seconds in Commodore's format. (Note— The RABBIT cannot be used to store data tapes from BASIC.)

The PET Rabbit is 2K of machine code supplied on cassette or in ROM. The cassette version occupies the top-most portion of memory and can be ordered in one of 5 locations: $1800-$1FFF for 8K PETs, $3000-$37FF or $3800-$3FFF for 16K PETs, and $7000-$77FF or $7800-$7FFF for 32K PETs. The reason for two different versions for the 16K and 32K PETs is to provide room for those programmers who use the DOS Support (wedge) program. (Note— The cassette RABBIT works only with 3.0 ROM PET's.)

The ROM version is a 24 pin Integrated Circuit which plugs into spare socket D4 and occupies memory $A000-$A7FF. Since the ROM version does not occupy user RAM, it will work with any 8K, 16K, or 32 K 3.0 or 4.0 ROM PET. The main advantage of the ROM Rabbit is that it doesn't have to be loaded each time you power up your PET and it does not occupy valuable RAM memory (4.0 ROM version at $9000).

The PET RABBITs high-speed cassette recording feature will not work with some of Commodore's older cassette decks. To be specific, cassette decks with the lift top lid (termed old style) will not work but all other features will work. In addition, we have discovered that some new style cassette decks will not work properly. How do you know if your cassette will work? Simple -open up the cassette deck and look at the printed circuit board components. If there are IC packages for all the active components, it will work with the RABBIT. If there are any transistors on the board, it will not work. Most new style cassette decks will work okay since there are very few of the transistor types. If you wish to purchase ROM RABBIT and a cassette deck, we can offer an attractive discount.

The RABBIT commands are:

SS	— Save with short leader	D	— Convert decimal # to hex #
SL	— Save with long leader	H	— Convert hex # to decimal #
L	— Load a program	Z	— Toggle character set
V	— Verify a program	K	— Kill the RABBIT
E	— Lcad and then run	*	— Go to monitor
T	— RAM memory test	G	— go to machine language program

RABBIT works with the PROGRAMMERS TOOL KIT.

CASSETTE RABBIT - $29.95
ROM RABBIT - $49.95
(specify memory, 3.0 or 4.0)

EASTERN HOUSE SOFTWARE
3239 Linda Dr.
Winston-Salem, N. C. 27106

U.S. Bank Check or International Money Order in U.S. dollars.

(LABEL), Y (LABEL,X) LABEL + INDX—1

LDA STA LDX STX LDY STY

SET IFM IFP IFN IFE

6502 Assembler/Editor

- APPLE
- ATARI
- PET
- KIM
- SYM

Before you buy that off-brand Assembler/Text Editor, note that EHS is the only company that provides a line of **compatible** ASM/TED's for the **PET/APPLE/ATARI/SYM/KIM** and other microcomputers.

When you make the transition from one of these 6502-based microcomputers to another, you no longer have to relearn peculiar Syntax's, pseudo ops, and commands. Not only that, EHS ASM/TED's are the **only resident 6502 Macro Assemblers** availiable and they have been available for several years. Thus you can be sure **they work.** Our ASM/TED's may cost a little more but do the others provide these **powerful features:** Macros, Conditional Assembly, String Search and Replace, or even up to 31 characters per label? Before you spend your money on that other ASM/TED, **write for our free detailed spec sheet.**

MACRO ASM/TED

- For APPLE/ATARI/PET/SYM/KIM
- Other than our MAE, no other assembler is as powerful.
- Macros/Conditional Assembly.
- Extensive text editing features
- Long Labels
- Designed for Cassette-based systems.

$49.95

MAE ASM/TED

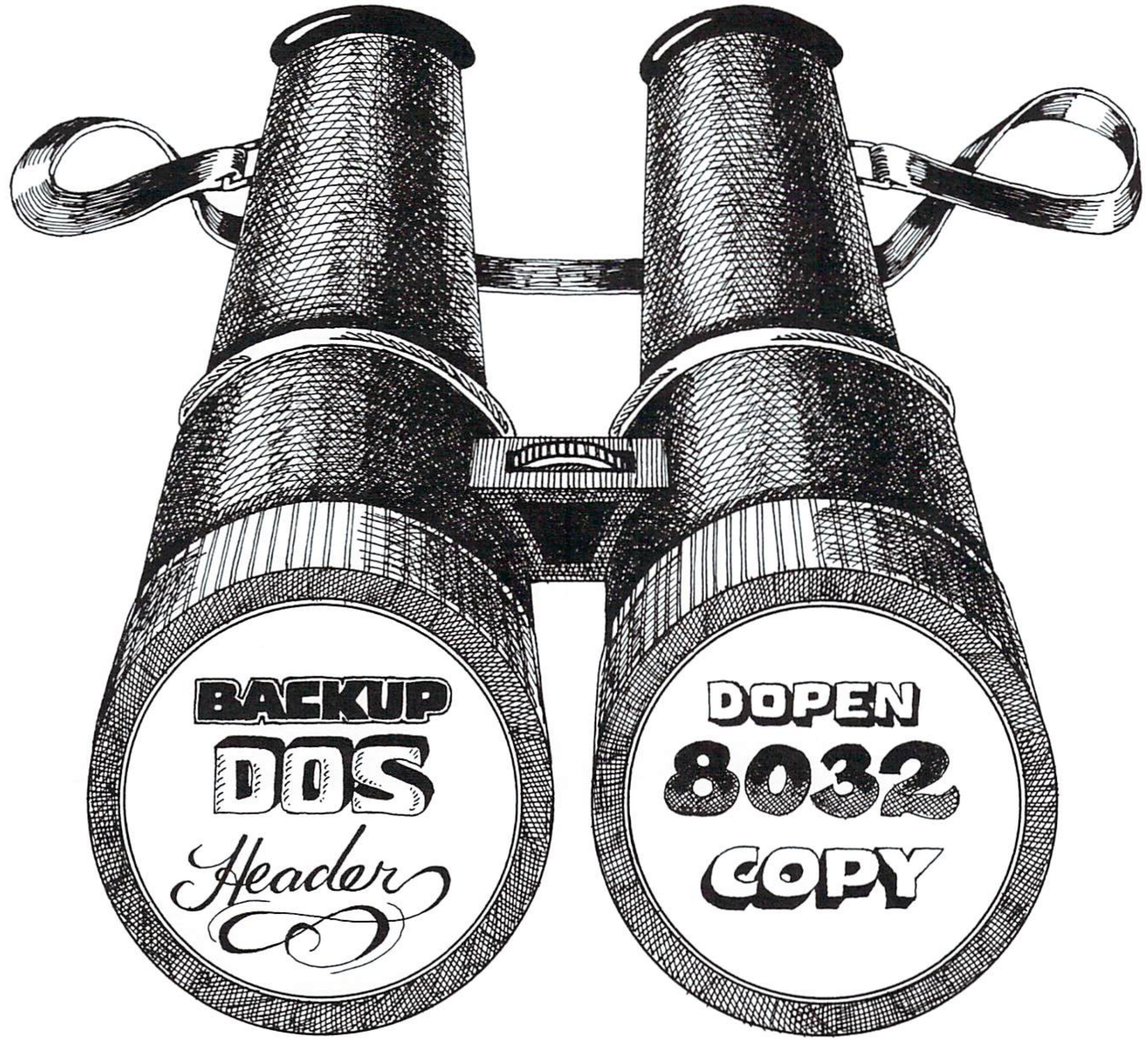
- For APPLE/ATARI/PET
- The most powerful ASM/TED
- Macros/Conditional and Interactive Assembly
- Extensive text editing features
- Long Labels
- Control files
- Designed for Disk-based Systems.

$169.95

EASTERN HOUSE SOFTWARE
3239 Linda Drive
Winston-Salem, N. C. 27106 USA
(Dealer Inquiries Invited)

PHONE ORDERS
(919) 924-2889
(919) 748-8446

.EN .BY .OS .BA .DE .CE



Some Observations On Using CBM 8032 & DOS 2.0

1. On closing files.
Sometimes your program bombs with a syntax error just after you have opened a disk file. And sometimes the active light on the drive stays on, even when you issue a DCLOSE. You're a little worried about it, since you want to start checking your syntax error, and don't want anything to happen to your file in the mean time.

Solution: Type direct command—OPEN 1,8,15:CLOSE 1, and the disk goes to sleep.

2. Spaces in files.
We all know now that leading blanks in a record are lost when INPUTting from a disk or tape file. If you need blanks at the beginning of a record, use shifted ones (CHR$(160))—the PET doesn't know they are blanks and gladly gives them back to you on INPUT. They PRINT on the screen as perfectly legal blanks of course.

3. Other nasties in files.
I am what you call a ''squasher'' of disk files. Whenever I get a chance, I'll ''code'' pieces of information so that they occupy as little space as possible. Example: suppose you have a series of Yes/No answers to be placed in a disk file. Let's say you have 8 of them, or less. You could use 8 bytes, one each with either a Y or a N. Or you can store this sequence of 8 answers in a single byte, by building a binary number out of the 8 answers, coding a Y as a 1, and a N as a 0. Below is a short piece of code to do it (we assume that the answers were input by your program into an array ANS$—i.e. ANS$(K)=''Y'' or ''N'', for K from 1 to 8):

```
2000  ANS%=0
2010  FOR I = 1 TO 8: IF ANS$(I)=''Y'' THEN
      ANS$+2**(8-I)
2020  NEXT I
```

The code constructs an 8-bit binary number that has a 1 in position i (leftmost position considered to be position 1) if ANS$(i)="Y", and a 0 in every other position. For example, NNNNYNYY will turn ANS% into 00001011, or decimal 11. We can now write this packed information to a file by PRINT# 1f, CHR$(ANS%). Isn't that great? Well, it is, provided we can unpack it easily, and provided we take a little care when writing it to the file.

I don't particularly like to do GETs from long records, so I want to get the information back via an INPUT statement. Assume that this ANS% byte is part of a longer string record (REC$), e.g. 20 character name, 8 Y/N answers, # of children, # of bathrooms in the dwelling.

Before proceeding, we should note that the record as described can be stored in 23 bytes, as long as we don't have more than 255 children or bathrooms. 255 is the biggest binary number that can be held in one byte (binary 11111111). We PRINT#1fn, NAME$; CHR$(ANS%); CHR$(#children); CHR$(#bathrooms) to our tape or disk file.

To get it back, we:

```
100  INPUT #1f, REC$
110  NAME$=LEFT$(REC$,20):
     ANS%=ASC(MID$(REC$,21,1)
120  CHILD=ASC(MID$(REC$,22,1):
     BATH=MID$(REC$,1)
130  REM: NOW TO ISOLATE OUR Y/N ANSWERS
     INTO ANS$
140  FOR K=8 TO 1 STEP -1: QZ%=ANS%/2:
     QR%=ANS%-2*QZ%
150  ANS$(K)="Y": IF QR%=0 THEN
     ANS$(K)="N"
160  ANS%=QZ%: NEXT K
```

So far, so good. HOWEVER—what happens if all the answers were N, or if the answers were NNNNYYNY, or if we had no children or thirteen bathrooms? What's special about these? They cause our program to become very confused, and sometimes result in the programmer becoming equally stumped. In all four of the above instances the program will either write a CHR$(0) or a CHR$(13) to the file, and nicely terminate the record in the wrong place. Both of those CHR$'s are recognized by the PET as end-of-record markers, and prevent us from getting to the rest of the record which follows the CHR$. What do we do about that? We make sure that they cannot occur as part of the file, except where we want them.

A few suggestions:

a. Start ANS% with a value other than 0, and which does not occur as one of the actual Y/N combinations. This is not feasible in our case of 8 Y/N's, since the entire byte ANS% is neded to cover the whole range of possibilities (from 0(00000000—all N's) to 255 (11111111—all Y's). In other words, there is no value with which ANS% can be initialized to avoid the "null or thirteen problem." Here we are forced to use GET instead of INPUT.

For sets of Y/N's of 7 or fewer, we can start ANS% at 2 to the power X, where X equals (# of Y/N answers):

Examples:

Nr of Y/N's	Starting value of ANS%
2	4
5	32
7	128

Let's look at 7 in more detail:

- if our answers are NNNNNNN, then ANS% will be 10000000 (binary), and CHR$(ANS%) cannot cause any problems when written to the file
- if our answers are NNNYYNY, ANS% equals 10001101, and again no record terminator is written.

On reading the file, we must of course remember that the value we read must now be reduced by the value we gave ANS% at the start. Line 110 in our short example will now read:

```
110  NAMES$=LEFT$(RECS,20):
     ANS%=ASC(MID$(REC$,21,1)-INIT
111  REM INIT IS STARTING VALUE OF ANS%
```

b. In our children-bathrooms example, the same ideas are of course valid. If we have N bathrooms, where N <128, we can add 128 to N before writing it to the file, and then subtract it from the value we read back before we actually use it in our program.

c. There is still one small thing to watch out for. When testing our program, we usually like to read stuff that we have written to a file, and display it on the screen, to make sure that what we thought we wrote did in fact get written. Watch out when you do this if you have compacted some of your data using the technique mentioned above. Many of the CHR$ characters that you write are non-printing ones—they don't appear on the screen on a PRINT command—in fact, they look like nulls (if you print them between two markers, say two *'s, the *'s will be printed as **, i.e. with nothing in between). Now that's OK as long as you are aware of this, and you don't have a CBM8032.

That beauteous machine grabs some of these CHR$'s, and interprets them as control codes, to define a screen window for example, or to ring its chimes. I recently printed out a large file to browse through the records, looking for abnormalities. The screen was scrolling merrily upward as the records were displayed, occasionally squeaking as a chr$(7) was printed. Suddenly the screen contracted, and the display was confined to the rightmost 10 columns of the

➡

screen. This was sort of cute and I could still read the data, although I had to hurry, since the short lines zipped by at a good clip. The next thing wasn't cute at all: the screen collapsed to a single character, at the bottom righthand corner, and the rest of my file just winked at me from that spot as it flashed by—still ringing the funny little bell at times.

Moral: When you want to look at "squashed" files, GET the characters one by one, and print out their ASC values. Make sure that you change any nulls to CHR$(0) prior to printing on the screen, otherwise your program will hang with an ILLEGAL QUANTITY ERROR—ASC("") is a nyetnyet.

4. The last word on nulls.
If you write a CHR$(0) to a file, and then GET it back (GET A$), then A$ does not equal CHR$(0), but rather comes back as a measly null character (""). This may not seem like much (eh!eh!), but it does become crucial when you want to copy one sequential file to another character by character—say because you processed part of it and now you want to discard that part and process only the remainder. During the copy, you must check if a character from the old file equals null ("") and change it to CHR$(0) before writing it to the new file—otherwise you end up with a mess.

5. Timeout error (variable ST=1)
I had never had one of those (at least not that I know of), and was therefore surprised and a bit pleased to finally get hit with one. Can you spot why in what follows? (FILE2 contains plenty of data)

```
10  OPEN 2,8,2"1:FILE2,S,R":OPEN
    3,8,3,"1:FILE3,S,W"
20  OPEN 1,8,15: REM DO A BUNCH OF STUFF
30  CLOSE 1
40  GET#2,A$:IF ST=64 THEN 60
50  PRINT#3,A$: GOTO40
```

Timeout occurred in line 50.

Note
Timeout is an IEEE condition indicating that a bus operation has taken too long to complete. The time allowed is 64 milliseconds and timeout can occur during a read or write operation. In the above example a timeout on write occurs at line 50; an attempt to send A$ across the bus is made but the DOS does not accept the character within the 64 ms. time limit. PET sets ST is set to 1 but no test is made to trap this error here.

Leaving this for a moment, another disaster lies in the OPEN1,8,15:CLOSE1 sequence of lines 20 and 30. As mentioned earlier, this causes the DOS to close down any open files on the disk but the 8032 still considers these files open since no CLOSE 2 or CLOSE 3 command was given. The GET# statement in line 40 is still honored (i.e. no FILE NOT OPEN ERROR) but the disk has nothing to offer since all files (in the DOS) are now closed. A timeout on read (ST=2) occurs but there is no BASIC to detect this (only ST=64? is tested).

The program then goes on to send the "gotten" character but (besides the fact that there is no character in A$ to send) there is no open file in the disk to send it to. The DOS won't accept A$ and after 64 ms., timeout on write is flagged (ST=1).

None of this should be of any concern if you handle your I/O files properly. Opening and closing the DOS command channel should not be used to close other R/W files except as mentioned earlier. By issuing proper OPEN and CLOSE commands, both computer and disk will always know what's happening, not to mention the programmer. Same goes for Basic 4.0 DOPEN and DCLOSE commands.

One last note, timeout on read (ST=2) will occur if you try reading past the end of a file. This usually happens when 'end of file' (ST=64) is tested after some subsequent bus operation is performed that changes ST. If this is a problem, the best solution is to "trap" ST into some other variable (say SX) and then test SX later.

Compressed Data
The above article talks about compressing data of the YES/NO, OFF/ON type so that several items can be stored as bits rather than using a whole byte for each. This can work well provided your program knows how to handle it. It can also save a lot of storage space when working with large amounts of information. The only drawback (aside from those mentioned) is that now your (user's) data is unreadable by other software, i.e. compressed data can look pretty alien to a simple file reading program, especially when compared to what was thought to be typed in. Compressing data is generally undesirable, but if you find yourself cramped for space and absolutely require it, always provide plenty of documentation. . .if not for the next guy, at least for yourself! ■

Sieg Delev
Kobetek Systems

Data Statements to Machine Code—A Disassembler

by Ed Steinfeld
Hudson, New Hampshire

Nearly every magazine I've picked up lately has had a PET program with part of the program neatly coded in DATA statements. It seems these programs are not compatible with the exact hardware or software I have. These data statements give me no clue as to what is being loaded into memory. All I do know is where it is loaded and how many bytes long the program is.

Well, this all finally came to an end. I had a program I really wanted but as usual it didn't work for my screen. First, I dumped it using the Monitor. That didn't give me too many additional clues to the program. Finally, I remembered a disassembly program a friend had started to write but never finished. After searcing about 20 disks I located it. Well it didn't work but it did have a fairly complete list of the MOS6502 mnemonics. With four-nights effort I was able to disassemble the progam and within minutes I had the program working and even modified.

I decided a disassembler program was a must and probably of use to others. Making it conform to the Commodore format wasn't an easy task. Putting the parantheses around the correct set of numbers and registers took some thought. The result is a program to disassemble memory contents into 6502 mnemonics, list the decimal location, hex location, hex contents of memory, display operands in decimal and print it all on a Commodore printer.

The DISASSEMBLER will work on any PET/CBM supporting a 2022 or 4022 printer. In the case of 40 or 80 column displays it will even center its name according to the size (BASIC V4 only). Systems without a Commodore printer will need all lines with a PRINT# modified, since the formatting feature of the CBM printer is used.

```
100 rem**"Disassembler  4 Oct, 1981"
110 rem**"Copyright (c) 1981"
120 rem**"E. F. Steinfeld
130 rem**"31 Richman Road
140 rem**"Hudson, NH 03051
150 rem**"Uses 40 or 80 col screen
160 if peek(213)=39thene=20:rem**"Looks for screen size"
170 print "▒"tab(34-e)"DISASSEMBLER"
180 print"Please wait while I set up..."
190 dim mn$(256),by%(256),co$(16)
200 for e=0 to 255
210 read mn$(e),by%(e)
220 next
230 fore=0to15
240 read co$(e)
250 next
260 sp$=chr$(29)
270 print"▒"chr$(21)
280 print:input "Start address (decimal)  0▌▌▌";ad
290 input"End address (decimal)  0▌▌▌";ae
300 print"Set printer to top of form.  Type any character when ready."
310 get ad$:if ad$=""goto310
320 open 4,4,1:open3,4:open 7,4,7:print#7
330 print#3,chr$(147)chr$(10)chr$(10)
340 open2,4,2:print#2,"  99999  aaaa  aaaaaaaa  aaaaaaaaaaaaa"
350 print#3,"Disassembly of adresses"ad"through"ae"
360 print#3
370 gosub460
380 print#3,spc(15)"*="ad$
```

→

```
390 goto530
400 sx=int(dc/16)
410 un=dc-(sx*16)
420 sx$=co$(sx)
430 un$=co$(un)
440 hx$=sx$+un$
450 return
460 hv=int(ad/256)
470  lv=ad-hv*256
480 dc=hv:gosub400
490 ad$=hx$
500 dc= lv:gosub400
510 ad$=ad$+hx$
520 return
530 ifae<=adthenprint#3,chr$(19):close4:close2:close3:end
540 ib=peek(ad)
550 mn$= left$(mn$(ib),3):iflen(mn$(ib))<4 thenv$="":goto570
560 v$=mid$(mn$(ib),4,1)
570 if mn$<>"nul"goto620
580 dc=ib:gosub 400
590 print#4,ad;sp$ad$sp$hx$+"*"
600 ad=ad+1
610 goto 530
620 on by%(ib) goto 640,700,870
630 rem**"Single byte instructions."
640 gosub460
650 dc=ib:gosub 400
660 print#4,ad;sp$ad$sp$hx$sp$mn$(ib)
670 ad=ad+1
680 goto 530
690 rem**"Two byte instructions."
700 gosub460
710 dc=ib:gosub 400
720 b1$=hx$
730 dc=peek(ad+1):gosub 400
740 b2$=hx$
750 op$=str$(peek(ad+1))
760 if v$=""thenmn$=mn$+op$
770 if v$=" "then mn$=mn$(ib)+op$
780 if v$="X"then mn$=mn$+op$+", X"
790 if v$<>"I"goto830
800 v$=mid$(mn$(ib),5,1)
810 if v$="X"then mn$=mn$+" ("+op$+", X)"
820 if v$="Y"then mn$=mn$+" ("+op$+"), Y"
830 print#4,ad;sp$ad$sp$b1$+" "+b2$sp$mn$
840 ad=ad+2
850 goto 530
860 rem**"Three byte instructions."
870 gosub460
880 dc=ib:gosub 400
890 b1$=hx$
900 dc=peek(ad+1):gosub 400
910 b2$=hx$
920 dc=peek(ad+2):gosub 400
930 b3$=hx$
940 op=peek(ad+1)+(peek(ad+2)*256)
```

```
 950 op$=str$(op)
 960 if v$="X" or v$="Y"then mn$=mn$+op$+", "+v$
 970 if v$="I" then mn$=mn$+" ("+op$+")"
 980 if v$=""then mn$=mn$+op$
 990 print#4,ad;sp$ad$sp$b1$+" "+b2$+" "+b3$sp$mn$
 1000 ad=ad+3
 1010 goto 530
 1020 rem**"Instruction mneumonic, number of bytes."
 1030 data "BRK",1,"ORAIX",2,null,0,null,0,null,0,"ORA",2
 1040 data "ASL",2,null,0,"PHP",1
 1050 data"ORA #",2,"ASL A",1,nul,0,nul,0,"ORA",3
 1060 data"ASL",3,nul,0,"BPL",2,"ORAIY",2
 1070 datanul,0,nul,0,nul,0,"ORAX",2,"ASLX",2,nul,0,"CLC",1,"ORAY",3
 1080 datanul,0,nul,0,nul,0,"ORAX",3,"ASLX",3,nul,0,"JSR",3,"ANDIX",2,nul,0
 1090 datanul,0,"BIT",2,"AND",2,"ROL",2,nul,0,"PLP",1,"AND #",2,"ROLA",1,nul,0
 1100 data"BIT",3,"AND",3,"ROL",3,nul,0,"BMI",2,"ANDIY",2,nul,0,nul,0,nul,0
 1110 data"ANDX",2,"ROLX",2,nul,0,"SEC",1,"ANDY",3,nul,0,nul,0,nul,0,"ANDX",3
 1120 data"ROLX",3,nul,0,"RTI",1,"EORIX",2,nul,0,nul,0,nul,0,"EOR",2,"LSR",2
 1130 datanul,0,"PHA",1,"EOR #",2,"LSRA",1,nul,0,"JMP",3,"EOR",3,"LSR",3,nul,0
 1140 data"BVC",2,"EORIY",2,nul,0,nul,0,nul,0,"EORX",2,"LSRX",2,nul,0
 1150 data"CLC",1,"EORY",3,nul,0,nul,0,nul,0,"EORX",3,"LSRX",3,nul,0,"RTS",1
 1160 data"ADCIX",2,nul,0,nul,0,nul,0,"ADC",2,"ROR",2,nul,0,"PLA",1,"ADC #",2
 1170 data"RORA",1,nul,0,"JMPI",3,"ADC",3,"ROR",3,nul,0,"BVS",2,"ADCIY",2,nul,0
 1180 datanul,0,nul,0,"ADCX",2,"RORX",2,nul,0,"SEI",1,"ADCY",3,nul,0,nul,0
 1190 datanul,0,"ADCX",3,"RORX",3,nul,0,nul,0,"STAIX",2,nul,0,nul,0,"STY",2
 1200 data"STA",2,"STX",2,nul,0,"DEY",1,nul,0,"TXA",1,nul,0,"STY",3,"STA",3
 1210 data"STX",3,nul,0,"BCC",2,"STAIY",2,nul,0,nul,0,"STYX",2,"STAX",2,"STXY",2
 1220 datanul,0,"TYA",1,"STAY",3,"TXS",1,nul,0,nul,0,"STAX",3,nul,0,nul,0
 1230 data"LDY #",2,"LDAIX",2,"LDX #",2,nul,0,"LDY",2,"LDA",2,"LDX",2,nul,0
 1240 data"TAY",1,"LDA #",2,"TAX",1,nul,0,"LDY",3,"LDA",3,"LDX",3,nul,0,"BCS",2
 1250 data"LDAIY",2,nul,0,nul,0,"LDYX",2,"LDAX",2,"LDXY",2,nul,0,"CLV",1
 1260 data"LDAY",3,"TSX",1,nul,0,"LDYX",3,"LDAX",3,"LDXY",3,nul,0
 1270 data "CPY #",2,"CMPIX",2
 1280 datanul,0,nul,0,"CPY",2,"CMP",2,"DEC",2,nul,0,"INY",1,"CMP #",2,"DEX",1
 1290 datanul,0,"CPY",3,"CMP",3,"DEC",3,nul,0,"BNE",2,"CMPIY",2,nul,0,nul,0
 1300 datanul,0,"CMPX",2,"DECX",2,nul,0,"CLD",1,"CMPY",3,nul,0,nul,0,nul,0
 1310 data"CMPX",3,"DECX",3,nul,0,"CPX #",2,"SBCIX",2
 1320 datanul,0,nul,0,"CPX",2,"SBC",2
 1330 data"INC",2,nul,0,"INX",1,"SBC #",2,"NOP",1,nul,0,"CPX",3,"SBC",3,"INC",3
 1340 datanul,0,"BEQ",2,"SBCIY",2,nul,0,nul,0,nul,0,"SBCX",2
 1350 data "INCX",2,nul,0,"SED",1
 1360 data"SBCY",3,nul,0,nul,0,nul,0,"SBCX",3,"INCX",3,nul,0
 1370 rem**"HEX conversion data."
 1380 data 0,1,2,3,4,5,6,7,8,9,"A","B","C","D","E","F"
 1390 end
ready.
```

PROGRAMMER'S TIPS

```
                 *=033A
826   033A   A9 00      LDA # 0
828   033C   8D 4C 03   STA 844
831   033F   A9 80      LDA # 128
833   0341   8D 4D 03   STA 845
836   0344   A0 05      LDY # 5
838   0346   BE 66 03   LDX 870, Y
841   0349   A9 66      LDA # 102
843   034B   9D C8 80   STA 32968, X
846   034E   CA         DEX
847   034F   D0 FA      BNE 250
849   0351   88         DEY
850   0352   30 11      BMI 17
852   0354   AD 4C 03   LDA 844
855   0357   18         CLC
856   0358   69 50      ADC # 80
858   035A   8D 4C 03   STA 844
861   035D   90 E7      BCC 231
863   035F   EE 4D 03   INC 845
866   0362   4C 46 03   JMP 838
869   0365   60         RTS
870   0366   0A         ASL A
871   0366   0F*
872   0366   14*
873   0369   19 1E 23   ORA 8990, Y
```

This is a disassembly of a program found on page 24 of INTERFACE, February 1981.■

MICRO-JOB (TM)

A COST ACCOUNTING SYSTEM

- For Professionals
  - Accountants/CPA's
  - Consultants
  - Attorneys
- For Corporations
  - Cost Accountants
  - Project Managers
- For Anyone who does Time and Material/Expense Accounting and Reporting

FEATURES

- User Defined Cost Components—You define the cost components or cost elements, e.g. labor, travel, and/or telephone, to which you will be posting hours and dollars.
- A Comprehensive User Manual.
- Current Period and To-Date Hours and Dollars are maintained for each job as well as an optional Customer/Client Number.
- Customer Lists as well as Customer Labels can be generated.

REPORTS

- Cost Component List
- Job List
- Job Detail/Summary Reports by Job showing:
  Cost Components
  Current Period and To-Date Hours and Dollars
- Cost Component Report by Cost Component showing:
  Job Numbers
  Current Period and To-Date Hours and Dollars
- Job Transaction Report by Job showing:
  Transaction Date
  Cost Component
  Transaction Description
  Hours and Dollars
- General Ledger Interface by Transaction showing:
  Transaction Date
  Job Number
  Cost Component
  Hours
  Transaction Description
  Account Number
  Dollars
  Offset (Balancing) Account Number
  Offset (Balancing) Dollars
- Period End Report showing:
  Opening Hours and Dollars
  Current Period Hours and Dollars
  Closing Hours and Dollars

$175.00

For a Free Brochure, send your name and address to:
The Financial Information Systems Corporation
P O Box 915
Manchester, CT 06040

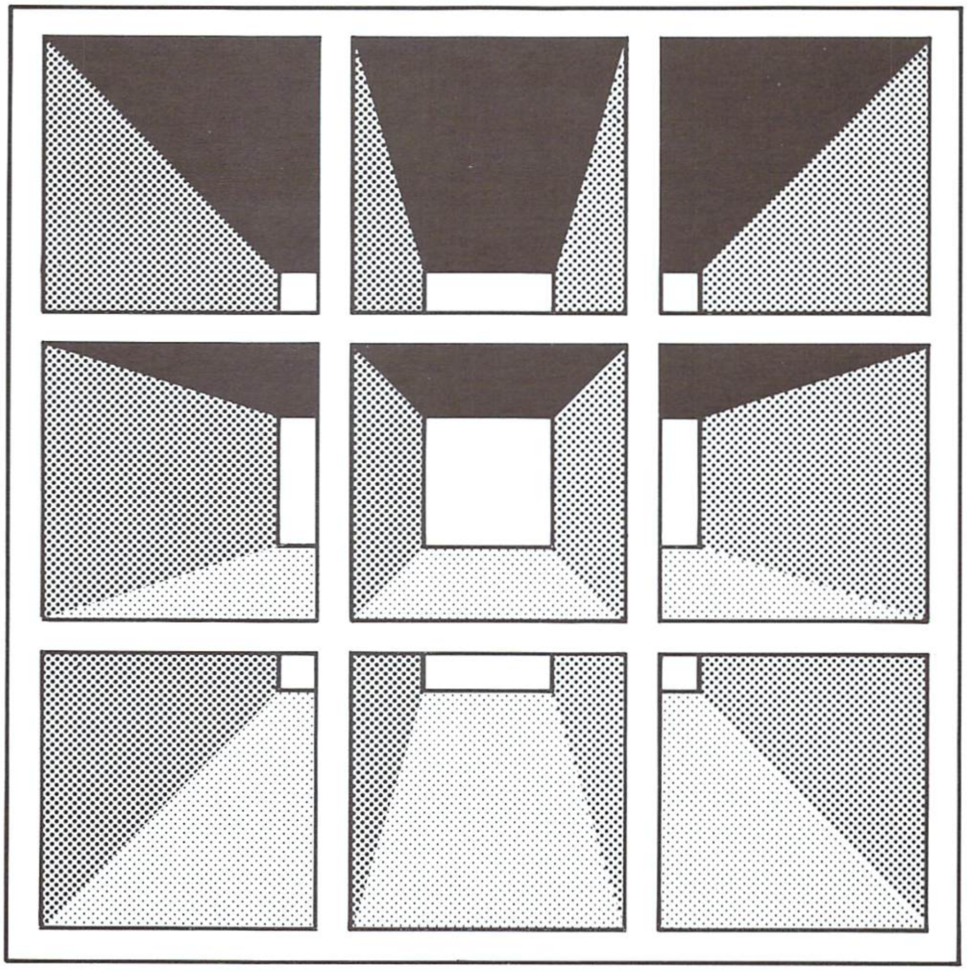
ATTENTION PROGRAMMERS

Commodore is compiling a list of software written for our computers. If you have software you would like to have included in this listing please submit the following for review:

- copy of program on disk or tape
- documentation describing purpose and utility of the program
- specify equipment necessary for program operation
- information on price of program and where the program can be purchased

Please submit this information to:

SOFTWARE Committee
Commodore Business Machines
681 Moore Rd.
King of Prussia, PA 19406



PET's VIEW OF ARRAY STORAGE

by
Elizabeth Deal
Malvern, PA

Over a year ago I had a memorable learning experience. I wanted to change the name of the array while a BASIC program was running but I didn't want the elements of the array to move, which is time and space consuming work on the PET. To do this I had to know where and how the operating system stores arrays. Once I knew that, the name change would be a snap. Right? Wrong. I read my manual, wrote a routine based on the book, and it never worked. Having spent hours trying to find a coding or logical error I was beginning to question my aptitude for POKEing two characters into the PET. Clearly, a fresh approach was called for.

A little listening to the PET proved the book incorrect, so the thing to do was to base a program on what the PET told me. The story has a happy ending and forms the background for this article.

Bear in mind that much of the time, you do not need to know this information, because the arrays take care of themselves nicely. But the day you need it, you'll be one step ahead of yourself. I also want to warn you that the material is confusing and boring. However, this article sets the record straight and shows a powerful method for solving similar problems you may encounter. You have been warned.

The data shown in this article was checked on three systems, UPGRADE PET (BASIC 2/3), 4032 PET (BASIC 4) and 80-column CBM (BASIC 4). Users of all PET/CBM releases would be well advised to check, and if necessary, correct their books. Three texts are mentioned in this article. Other books should also be checked as information of this sort seems to travel far and linger forever.

Review

Let's briefly review how arrays are stored in the PET with emphasis on the array header information and storage of character string arrays.

The BASIC interpreter puts arrays adjacent to single variables. The bounds of array storage are pointers in 44-45 to 46-47 ($2C-$2D to $2E-$2F). Integer arrays (V% type)

➡

occupy two bytes per element, floating point arrays (V type) occupy five bytes per element. The elements of those two types are stored within this area. The actual storage scheme is fully and adequately shown in the texts cited below.

Character string arrays (V$ type) occupy three bytes per element within the array storage area, hereafter referred to as declaration or parameters. Byte #1 holds the length of string, byte #2-low byte and byte #3-high byte of element's address, plus all the room, usually near the screen memory, that is required to hold the actual character strings. (Some texts reverse bytes 2 and 3, making it impossible to find an array element). Each element of the string array is placed in high memory in backward order.

The area occupied by each element depends on the length of the string. If A$(2)="12345" then five bytes are required. This description is correct in the books. Basic 4 systems store character strings under the same protocol, but additionally, two bytes of a pointer are stored next to each element. Hence, our A$(2) in Basic 4 requires seven bytes. The most recent Osborne guide does not reflect this fact.

In addition to that memory allocation, each declared or implied array occupies several "overhead" bytes. The overhead immediately precedes the elements of the array in case of math arrays, and it precedes the parameters of string arrays. The header describes the attributes of the array: its name (in coded form that specifies type), how much memory it occupies, how many dimensions it has and the size of each dimension (zero element included). Though the texts describe the contents of the overhead area correctly, the details are the weak spot. The facts as I see them are shown in Figure 1 (one).

When in Doubt, Ask Your PET

There is a small BASIC routine in both Osborne PET/CBM guides which permits you to look at the array storage scheme. The output is in decimal and if you read the array name coding system you should have no trouble seeing how arrays are handled by BASIC. The routine is on page 325 in book 1 (red) and on page 348 in book 2 (white).

Alternately, if you are comfortable using the Monitor you can see the information directly in a nice compact form, all neatly aligned before you. If you are not comfortable with the monitor, this is your chance to learn. The Monitor is handy, its use not as error prone as long BASIC expressions and it can supplement BASIC work, as needed. I can't think of one good reason why we shouldn't use the Monitor in this sort of work.

It helps to know a little about hexadecimal numbers. But it is not necessary, since we are mostly interested in patterns and landmarks rather than the decimal equivalents that BASIC uses. I will explain terms as we go along digging. Cold start your computer, and do not load the DOS support if you want your numbers to be the same as in the illustrations. To enter the monitor, type SYS4 or SYS1024 and RETURN. If you want to use a printer, then in Basic 4 system you must type SYS54386 instead. Then type M followed by what you see in Figure 2. As in BASIC, always RETURN each line. To leave the Monitor and re-enter Basic, type X and RETURN.

Here is what I saw in the 32K Upgrade PET, 4032 PET and the 80-column CBM after RUNning a one line Basic program which simply initializes the overhead area and fills the element area with zeros.

**** FIGURE 1 ****

Overhead Byte #	Contents
1- 2	Two character coded array name
3- 4	Memory used for the array and N-byte overhead, stored low order byte first
5	Number of dimensions
6- 7	
8- 9	
10-11	As many pairs as there are dimensions:
12-13	These pairs indicate size of each dimension in reverse order from that in the DIM statement
.	and are stored high order byte first.
.	
- N	

```
 10 DIM AA(259),AA%(3,257),AA$(5,6,7,8)
READY.
RUN

READY.
SYS4

B*
      PC  IRQ  SR AC XR YR SP
.;  0005 0345 30 00 5E 04 F8
.M 002C-002C
.:  002C 29 04 DA 34 00 80 EC 7F
.M 0429 0431
.:  0429 41 41 1B 05 01 01 04 00
.:  0431 00 00 00 00 00 00 00 00
.M 0944 094C
.:  0944 C1 C1 19 08 02 01 02 00
.:  094C 04 00 00 00 00 00 00 00
.M 115D 1165
.:  115D 41 C1 7D 23 04 00 09 00
.:  1165 08 00 07 00 06 00 00 00
./
```

$41 is hex code for A. $C1 is hex code for A plus $80 (128 decimal) according to the type-coding protocol correctly described in the Osborne and Commodore books.

Let's find some landmarks, in hex. Use a pencil on the illustrations, or, better yet, your cursor on the screen location you are reading about. If you have trouble counting, the hex system goes like this:...9, A, B, C, D, E, F, 10, 11....And do use your fingers. What's 7 bytes beyond $0429? 0429...A, B, C, D, E, F...0430.

Remember, we don't really care what these numbers mean in decimal, we just want to put some markers on things. The array area begins at $0429, ends at $34DA. $34 is the high byte (page, or most significant part of the number) and $DA is the low byte of the address. Array AA begins at $0429 and uses up $051B-1 bytes (one less than $051B). Array AA% begins at $0429+$051B=$0944 and uses up $051B-1 bytes, leading us to the last array, AA$, which begins at $1159 and occupies $237D−1 bytes.

As far as I can tell this is precisely how the PET finds an array when you refer to it in your BASIC program. The PET hops from one to the next until the name matches. The size of the skip (search increment) depends on how many bytes an array occupies.

Don't forget that the first element of the array is called zero and not one. Therefore DIM AA$(5,6,7,8) is a six by seven by eight by nine array. Note, in Figure 2, that the size (# elements) of last dimension is first and first is displayed last, like this: 09 08 07 06. In case of the one dimensional A(259) the PET says "size of dimension is 260," which breaks down to a high byte of INT(260/256)=1 and low byte of 260−256*high=4. Hence, you see 01−04 sequence in bytes 6 and 7 of the header (this is an example of high order byte first).

You should be able to see that if you reverse the high-low byte order in address or size of the array, the probability of finding an array drops to zero. And that is precisely what caused trouble and triggered this whole story with me.

Character String Arrays

A note to 16K PET users: the following section will discuss adresses at the top of 32K PET's memory ($8000). Whenever you see the most significant part of the address being $7F, as in $7FE9, substitute $3F.

RUNning the two-line program on the Upgrade (Figure 3) and Basic 4 (Figure 4) systems reveals that the character string arrays declaration or overhead is in the order of strings appearing in the program. The area occupied by header and parameters is 5 lines or 40 bytes long ($28). Parameters (length and address) of the first element, C$(0), are followed by parameters of the second element C$(1) and so on until the last, and eleventh element, C$(10). Both Osborne books incorrectly describe this information.

The elements of the array are stored backwards in memory. In the Upgrade PET, one finds that C$(10) is in $7FE9, C$(0) is in $7FFE. $E9 is smaller than $FE. Sequence 20-31-30 means space, one, zero, meaning " 10", sequence 20-39 means " 9", 20-30 means " 0". The printout for the Upgrade system reflects this order:

```
 10 DIM C$(10)
 20 FORI=0TO10:C$(I)=STR(I):NEXT
READY.
RUN

READY.
SYS4

B*
      PC  IRQ  SR AC XR YR SP
.;  0005 0345 30 00 5E 04 F8
.M 002C 002C
.:  002C 30 04 58 04 E9 7F EC 7F
.M 0430 0457
.:  0430 43 80 28 00 01 00 0B 02
.:  0438 FE 7F 02 FC 7F 02 FA 7F
.:  0440 02 F8 7F 02 F6 7F 02 F4
.:  0448 7F 02 F2 7F 02 F0 7F 02
.:  0450 EE 7F 02 EC 7F 03 E9 7F
.M 7FE9 7FFF
.:  7FE9 20 31 30 20 39 20 38 20
.:  7FF1 37 20 36 20 35 20 34 20
.:  7FF9 33 20 32 20 31 20 30 20
./
```

➡

The Basic 4 storage order is identical to the Upgrade system. But since each element occupies two extra bytes, the addresses of the elements are different. This shows up in Figure 4. Note that the illustration differs slightly from previous illustrations which were screen dumps. Figure 4 is a printout and does not show my end of the conversation. It only shows PET's amswers. My questions were

M 002C 002C
M 0430 0457
M 7FD3 8000 (7FD3 is picked up from the tail end of line 0450)

```
  10 DIM C$(10)
  20 FORI=0TO10:C$(I)=STR$(I):NEXT
 READY.
 C*
      PC   IRQ   AC  XR  YR  SP
 .;  B780 E455  34  33  38  36  FA
 .
 .:  002C 30 04 58 04 D3 7F D6 7F
 .
 .
 .:  0430 43 80 28 00 01 00 0B 02
 .:  0438 FC 7F 02 F8 7F 02 F4 7F
 .:  0440 02 F0 7F 02 EC 7F 02 E8
 .:  0448 7F 02 E4 7F 02 E0 7F 02
 .:  0450 DC 7F 02 D8 7F 03 D3 7F
 .
 .
 .:  7FD3 20 31 30 55 04 20 39 52
 .:  7FDB 04 20 38 4F 04 20 37 4C
 .:  7FE3 04 20 36 49 04 20 35 46
 .:  7FEB 04 20 34 43 04 20 33 40
 .:  7FF3 04 20 32 3D 04 20 31 3A
 .:  7FFB 04 20 30 37 04 20 31 30
 .
 .
 READY.
```

To repeat: The parameters of the string arrays are stored in order, the actual elements of the string array are stored in reverse order. The text and diagrams in both Osborne books need correction.

I have no reason to suspect that other PET/CBM systems will give information different from that presented here.

Time to Fix the Books
If you use those texts you may want to check and correct the specific pages shown below. Similarly, if you use other texts, you might be able to square the information with what you have just read, to see if they also need fixing.

1. Upgrade PET manual (CBM 2001-16, −32, 3016, 3032 Professional Computer User Manual, June 1979, P/N 320856-3), page A-6. Some pointers are relevant to the original ROM PET, and irrelevant to the upgrade PET. The manual suggests that to search through an array table one needs to increment the search pointer by the contents of byte 3 of the header (3-4 is correct). I suspect that this error got into the book because there existed a 255-element limitation in the original ROM PET. This limitation has been lifted with the introduction of the Upgrade systems.

2. Osborne book 1, red, (PET/CBM Personal Computer Guide, Donahue and Enger, 1980) presents confusing information. The header information in diagrams on page 325 is correct. Page 322 contains an incorrect diagram and page 323 incorrectly describes bytes 5 to N. Page 323 incorrectly specifies that the address of the character string element is stored high order byte first. The string area diagram on page 325 is incorrect, as the elements are stored backwards in memory, not in declaration. C$(10) etc. subheadings are reversed. Strings A-K are, in fact, stored in K-A order.

3. Osborne book 2, white, (PET/CBM Personal Computer Guide, Osborne and Donahue, 1980) is more consistent. Relevant information on pages 347, 348 and 349 is wrong. Description of bytes 3 to N is wrong. This book suggests that the storage needed for the array is in byte 4. It is in fact in 3 and 4. The diagram of character string storage on pages 345 and 348 is incorrect in that the pointer to the array element is in fact stored low order byte first. The arrays storage description on page 349 is incorrect in the same respect as in book 1.

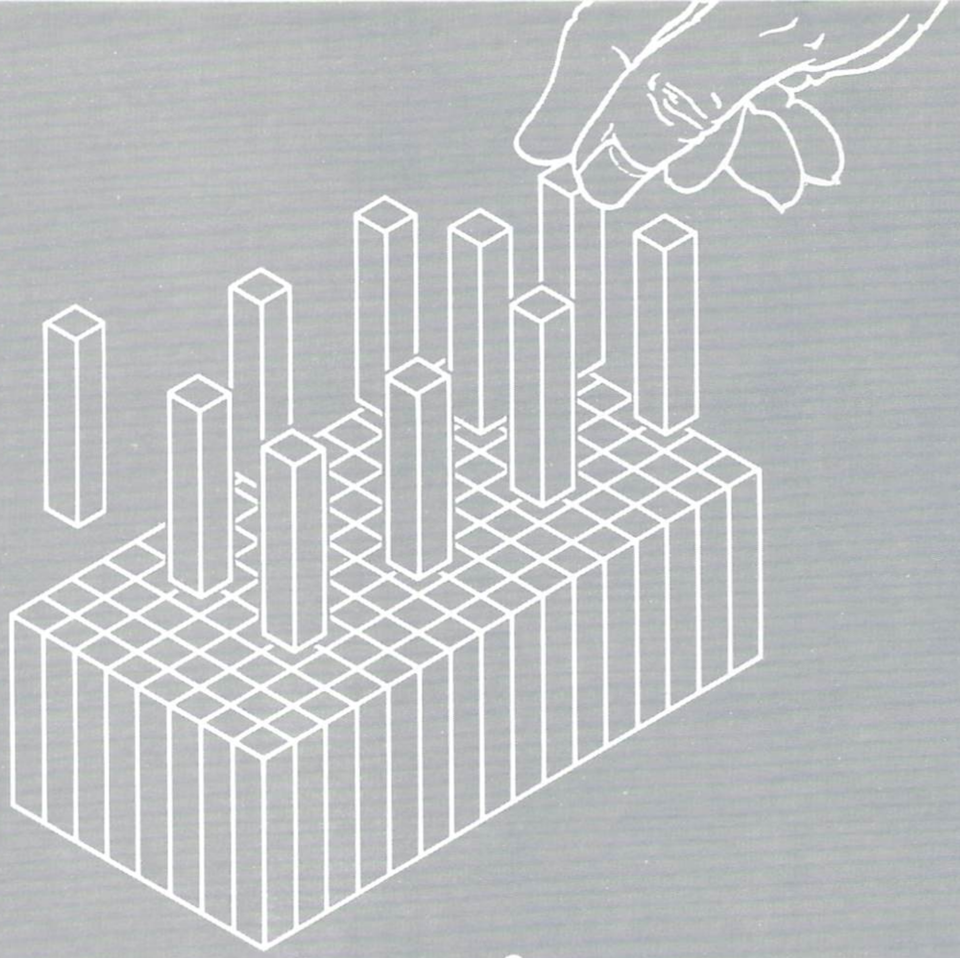
Finally, the string array information in this book is VALID ONLY for UPGRADE ROM PET/CBMs. The Basic 4 declaration section and the actual storage section do not reflect the fact that each element of string array occupies two more bytes of storage than in the Upgrade systems.

References:
Figures 2 and 3 are screen dumps using an UPGRADE PET KEYPRINT routine written by Charles Brannon and published in Nov./Dec. 1980 COMPUTE!

The inspiration for digging comes from reading materials published by Jim Butterfield, specifically the memory maps without which work could get very difficult.

I am grateful to COMPUTER FORUM of Frazer, PA for permitting me to use their Basic 4 equipment for printouts and to Pat McAllister of COMMODORE—Pennsylvania for telling me the results of the character string program in a Basic 4 PET.■

© *Elizabeth Deal*



Easy Choice

The TOOLKIT™ is one of the most widely used enhancements for the PET/CBM microcomputer. This collection of programming aids makes writing, debugging and polishing BASIC programs easier - so easy that these 9 "tools" have become an integral part of the PET microcomputer.

Ask for it by name - See your local Commodore dealer.

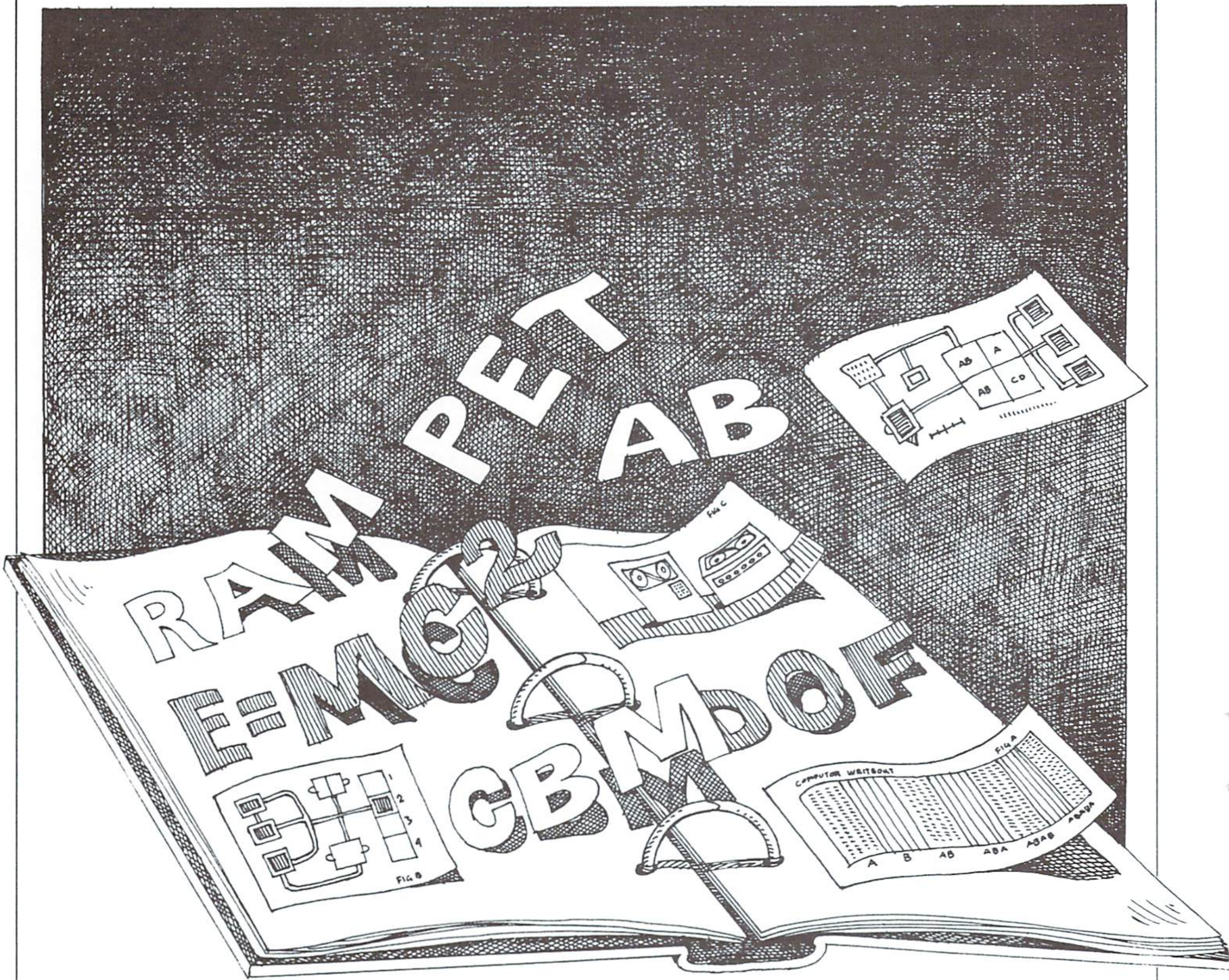
AUTO	your PET will automatically number lines so you don't have to manually enter this information.
DELETE	every line within the range of numbers will be instantly removed from your program without the tedious process of entering line numbers and pressing return.
HELP	simply by typing "help" you will receive immediate information as to where your error occurred.
RENUMBER	this feature allows you to change all line numbers and all references to those numbers.
TRACE	now you can see the exact order and sequence of your program's execution.
STEP	like TRACE, but one statement at a time can be executed.
DUMP	during or after running a program, your PET will display the names and values of all variables used in your program.
FIND	this command allows you to zero-in on only those lines which contain the desired character string.
APPEND	adds a program previously saved on tape to your current program.

The original only $39.95!

BASIC Programmer's TOOLKIT™

PALO ALTO IC's

2585 East Bayshore Rd., Palo Alto, California 94303 (415) 493-8665



Excerpts from a Technical Notebook

DOS 2.1 and 2.5 Bugs

The following bugs are to be found in both DOS 2.1 and 2.5:-

1. Disk Full Message Too Late
The disk operating system is supposed to check the number of blocks still free on the disk. When only two remain, the 'Disk Full' error message is supposed to be sent and the file closed, thus retaining some of the information.

What actually occurs is that the 'Disk Full' message is sent when there are no free blocks on the disk. The file cannot be closed properly. The BAM is updated to show that there are no free blocks and the file shows the number of blocks written to disk as zero. The file is also tagged as being unclosed by the use of an asterisk. A 'COLLECT' command has to be used to clear the files from disk and recover the lost blocks.

Using a save with replace causes the disk to write the new information to an unused area of the disk, close the file, and scratch the old version. When the 'Disk Full' message error is sent, output terminates and the new file does not show on the disk. The BAM is updated to show zero blocks free. The old file which was to be replaced remains untouched. A COLLECT is needed to recover the used blocks by the unclosed file.

There is no simple manner in which the lost data can be recovered and it is also difficult to test the disk before writing any information out to it without knowing the exact size of the required file, the simplest solution is to keep a check on the amount of free space with a catalogue request and start a new disk when space starts to run short.

2. Block/Memory Commands Destroys Error Information.
Since the block and memory commands use the error channel for communicating with DOS, the last information in the error channel is always destroyed. This is not a bug. Any error checks to and from the disk should always be made straight after the disk command has been given. On the 4000/8000 series use DS for the error code and DS$ for the error message.

3. Calling The Directory From Both Drives.
If no drive is specified, the DOS returns the directory from the last drive which was accessed and then the second directory. The drive from which the directory has come is shown next to the disk title.

4. Load/Run.
Drive zero initializes and the first file is read in. If the file is not a program, an error message is generated. If the drive has already been used then the last program called up will be read into the machine. If you wish to use the quickload feature (Shift RUN/STOP), ensure the program to be run is thefirst on the directory.

5. Copying Files.
If the user sends more than four files to be copied from one drive to another, the DOS should return an error message. Under some circumstances this does not occur and the error message 'Disk ID Mismatch' will be given. Do not send more than four lines to be copied at a time. For large amounts of copying use the disk utility 'copy disk files.'

6. Copying Files on the 8050.
The copy command on the 8050 will sometimes fail after eight copies when 'COPY DO TO D1' is given, resulting in 'Illegal Track for Sector' or 'Disk ID Mismatch' as the error message. Copy also fails after eight copies when performing matched copying using 'COPY DO, "TEST*" TO D1,"*".

The copy bug is not present on DOS 2.1 as the two systems are different. (DOS 2.5 has to keep bringing in the BAM for the drives while DOS 2.1 keeps both BAM's in the DOS RAM.)

7. Block Allocate on the 8050 and 4040 Disk Systems.
The Block Allocate (B-A) command is for use with the random access commands so that the disk unit knows that the block has been used and cannot be allocated to programs or sequential files. On the 3040 and 4040 drive the BAM on the disk is updated only when a file is closed. The drive should not be initialized before the BAM is written back to the disk as this actually calls the old BAM from the disk. B-A is supposed to report via the error channel whether a block has been allocated.
If the block has not been used it reports OK but if it has then the message 'NO BLOCK' is given. The error message also reports the next free track and sector. On the 3040/4040 this works correctly. On the 8050 there is a bug in DOS which can cause problems. For example, if the whole of track one has been used then DOS will report 'NO BLOCK 1 21'. This is obviously incorrect in that no sector 21 exists, i.e., DOS reports an illegal sector and will not search other tracks. Thus, if a program requires the next block it has to perform a track by track search and test for illegal sectors to determine when it has found a free block.

8. Disk Initialization on DOS 2.5.
There is a problem with the 8050 disk system at power on. The bug has the appearance of being sporadic but is very repeatable when the cause is known. For instance, the disk may not initialize, programs will not be found or the directory will not be displayed. The problem is caused by having the read head above track 55 at power on. The disk system can not recognize the problem and tries to read information from the track that it is on causing considerable confusion.

In business systems where a shift run is the method of starting the program, it is important that the head be moved to around track 39. This can be done during the system power-down at the end of the previous day, by performing an initialize via the error channel. The disk will then be in the correct state for use the next day. Note that the problem only occurs when the disk is powered down and then up again.

9. Disk Initialization DOS 2.1.
DOS 2.1 has an auto-initialize routine which relies on the disk ID changing. Thus, it is important for the user to ensure that IDs are different between disks or that the initialize command is given via the error channel. DOS 2.5 has a microswitch which will initialize the disk whenever it is opened and closed.

10. Block Write.
This command should not be used under any circumstances on DOS 2.1 because it clogs up the error channel. The only solution is to reset the disk by power down. Use the function U2 instead.

11. Disk Status DS$.
The disk error string does not monitor the commands which utilize the error channel (15). The commands associated with this are all the low-level block and memory commands such as Block-Read. The solution is to read the error from disk using:

```
10 INPUT£15,A$,B$,C$,D$,: PRINT A$,B$,C$,D$
```

12. Channel 14.
There is no channel 14 in DOS 2.x, use only those between 2-13 for normal random access. Secondary addresses 0 and 1 are used for LOAD and SAVE and should only be used under special circumstances.

Differences between old and new 4016/32s

There are potential software problems arising from the recent introduction of the 12" screen 4032 and the impending introduction of the 12" screen 4016.

Values returned by PEEK (151)
Both the 9" and the 12" screen machines use $97 (decimal 151) to hold a value related to the key being depressed. In the 9" 4016/32 this value relates to the keyboard matrix. In the 12" 4016/32 it is pseudo-ASCII (i.e. the 9" is like the 3032, while the 12" is like the 8032).

➡

4016/32 Keyboard layout

15	80	72	79	71	78	70	77	69	76	68	75	7	14	74	66	73	65
@	!	"	#	$	%	'	&		(	)							
64	33	34	35	36	37	39	38	92	40	41	95	91	93	19	17	29	20
8	64	56	63	55	62	54	61	53	60	52	59	5	12	58	50	57	49
	Q	W	E	R	T	Y	U	I	O	P		<	>	7	8	9	/
18	81	87	69	82	84	89	85	73	79	80	94	60	62	55	56	57	47
	48	40	47	39	46	38	45	37	44	36		27		42	34	41	33
	A	S	D	F	G	H	J	K	L	:				4	5	6	*
	65	83	68	70	71	72	74	75	76	58		13		52	53	54	42
	32	24	31	23	30	22	29	21	28	20				26	18	25	17
	Z	X	C	V	B	N	M	,	;	?				1	2	3	+
	90	88	67	86	66	78	77	44	59	63				49	50	51	43
					6									10	2	9	1
														0	.	-	=
					32									48	46	45	61

The figure above each character is the value returned at $97 on the 9" screen 4016/32; the figure below is that on the 12" screen 4016/32.

To check whether a program uses PEEK(151) use the FIND command in the Basic Programmers Toolkit or Basic Aid.

Top of second cassette buffer
40-column machines use 224-248 ($E0-F8) for a table of screen wrap-round flags. Because such a table became unnecessary in 80-column machines, these addresses were used for other purposes (E.G. bell, repeat, keyboard buffer length). The 12" 4016/32 has these 8032 features, but needs the table of wrap-round flags. Therefore the 12" 4016/32 uses even more of the top of the second cassette buffer than does the 8032. No programs should now use addresses between 1001 and 1024 ($03E9-0400).

Memory Map for top of second cassette buffer in 12" 4016/4032

12"	4016/32	8032		
03E9	1001	00E6	230	Repeat Key Delay Counter
03EA	1002	00E5	229	Delay between Repeat Counts
03EB	1003	00E3	227	Maximum Keyboard Buffer Size
03EC	1004	00E7	231	Bell Enable and Delay Count
03ED	1005			50 Herz Jiffy 5 Counter
03EE	1006	00E4	228	Repeat Key Flag
03EF	1007			Mask for Tabs
03F0	1008			
03F1	1009			
03F2	1010			
03F3	1011			
03F4	1012			10 bytes for Tabs
03F5	1013			
03F6	1014			
03F7	1015			
03F8	1016			
03F9	1017			
03FA	1018			User command for Monitor
03FB	1019			
03FC	1020			Timeout on IEEE
03FD	1021			
03FE	1022			
03FF	1023			
0400	1024	0400	1024	Start of Basic

Commodore ROM Genealogy

When the PET-2001 first went into production in September 1977 there were two ROM sets incorporated into the system, known as BASIC 1.0. One set was the 6540 28-pin ROM by MOS Technology Inc. and the other was the 2316 24-pin ROM.

The next upgrade production was with two ROM sets known as BASIC 2.0. These corrected an intermittent bug in the edit software and improved the garbage collection routines.

The next two production ROM sets are generally known as BASIC 3.0. This upgrade allowed interfacing to the Commodore disk system. It also cleared up a bug limiting the dimensions of arrays and improved the garbage collection.

Also, at this time the CBM Professional Computer came into being. One set of ROMs was for the Graphic (PET) keyboard and the other was for the Business CBM.

The next upgrade known as BASIC 4.0 added the Disk

Commands to ROM and greatly improved the garbage collection. This has been further upgraded to BASIC 4.1 to correct errors in version 4.0. At the same time Commodore brought out the new 80-column machine (the 8032) with an enhanced screen editor.

There have been three different Character Genrerator ROMs installed over these generations. Earlier production runs through BASIC 2.0 had the 6540-010 (p/n 901439-08) and 901447-08 (p/n 901447-8). BASIC 3.0 and 4.0 used the 901447-10 (p/n 901447-10).

The 901447-10 ROM can replace the 901447-08 ROM in up-grading from BASIC 2.0 to BASIC 3.0. There is no replacement ROM for the 6540-010 28 pin ROM.

The 2022 (tractor feed model) and 2023 (friction feed) printers were discontinued in 1980.

ROM 1.0 — Basic Level I
28 pin ROM type 6540 — Series 2001

Location	ROM #	Part Number
H1	6540-011	901439-01
H2	6540-013	901439-02
H3	6540-015	901439-03
H4	6540-016	901439-04
H5	6540-012	901439-05
H6	6540-014	901439-06
H7	6540-018	901439-07
A2	6540-010	901439-08

ROM 1.0 — Basic Level I —
24 pin ROM type 2316B — Series 2001

Location	ROM #	Part Number
H1	901447-01	901447-01
H2	901447-03	901447-03
H3	901447-05	901447-05
H4	901447-06	901447-06
H5	901447-02	901447-02
H6	901447-04	901447-04
H7	901447-07	901447-07
A2	901447-08	901447-08

ROM 2.0 — Basic Level II -
28 pin ROM type 6540 — Series 2001

Location	ROM#	Part Number
H1	6540-019	901439-09
H2	6540-013	901439-02
H3	6540-015	901439-03
H4	6540-016	901439-04
H5	6540-012	901439-05
H6	6540-014	901439-06
H7	6540-018	901439-07
A2	6540-010	901439-08

ROM 2.0 — Basic Level II —
24 pin ROM type 2316B — Series 2001

Location	ROM #	Part Number
H1	901447-09	901447-09
H2	901447-03	901447-03
H3	901447-05	901447-05
H4	901447-06	901447-06
H5	901447-02	901447-02
H6	901447-04	901447-04
H7	901447-07	901447-07
A2	901447-08	901447-08

ROM 3.0 — Basic Level III —
28 pin ROM type 6540 — Series 2001

Location	ROM #	Part Number
H1	6540-020	901439-13
H2	6540-022	901439-15
H3	6540-024	901439-17
H4	6540-025	901439-18
H5	6540-021	901439-14
H6	6540-023	901439-16
H7	6540-026	901439-19
A2	6540-010	901439-08

ROM 3.0 — Basic Level III —
24 pin ROM type 2316B — Series 2001

Location	ROM #	Part Number
H1	901465-01	901465-01
H2	901465-02	901465-02
H3	901465-24	901465-24
H4	901465-03	901465-03
H5	Blank	
H6	Blank	
H7	Blank	
A2	901447-08	90147-08

ROM 3.0 — Basic Level III —
Large Graphic Keyboard — Series 2001

Location	ROM #	Part Number
D3	Blank	
D4	Blank	
D5	Blank	
D6	901465-01	901465-01
D7	901465-02	901465-02
D8	901465-24	901465-24
D9	901465-03	901465-03
F10	901447-10	901447-10

ROM 3.0 — Basic Level III —
Business Keyboard — Series 2001

Location	ROM #	Part Number
D3	Blank	
D4	Blank	
D5	Blank	
D6	901465-01	901465-01
D7	901465-02	901465-02
D8	901474-01	901474-01
D9	901465-03	901465-03
F10	901447-10	901447-10

→

ROM 4.0 — Basic Level IV — Graphic Keyboard — Series 2001 & 4000

Location	ROM #	Part Number
D3	Blank	
D4	Blank	
D5	901465-19	901465-19
D6	901465-20	901465-20
D7	901465-21	901465-21
D8	901447-29	901447-29
D9	901465-22	901465-22
F10	901447-10	901447-10

ROM 4.0 — Basic Level IV — Business Keyboard — Series 2001 & 4000

Location	ROM #	Part Number
D3	Blank	
D4	Blank	
D5	901465-19	901465-19
D6	901465-20	901465-20
D7	901465-21	901465-21
D8	901474-02	901474-02
D9	901465-22	901465-22
F10	901447-10	901447-10

ROM 4.1 — Basic Level IV - Graphic Keyboard — Series 2001 & 4000

Location	ROM #	Part Number
D3	Blank	
D4	Blank	
D5	901465-23	901465-23
D6	901465-20	901465-20
D7	901465-21	901465-21
D8	901447-29	901447-29
D9	901465-22	901465-22
F10	901447-10	901447-10

ROM 4.1 — Basic Level IV — Business Keyboard — Series 2001 & 4000

Location	ROM #	Part Number
D3	Blank	
D4	Blank	
D5	901465-23	901465-23
D6	901465-20	901465-20
D7	901465-21	901465-21
D8	901474-02	901474-02
D9	901465-22	901465-22
F10	901447-10	901447-10

ROM 4.0 — Basic Level IV — Series 8000

Location	ROM #	Part Number
UD6	901465-22	901465-22
UD7	901474-03	901474-03
UD8	901465-21	901465-21
UD9	901465-20	901465-20
UD10	901465-19	901465-19
UD11	Blank	
UD12	Blank	
F10	901447-10	901447-10

ROM 4.1 — Basic Level — Series 8000

Location	ROM #	Part Number
UD6	901465-22	901465-22
UD7	901474-03	901474-03
UD8	901465-21	901465-21
UD9	901465-20	901465-20
UD10	901465-23	901465-23
UD11	Blank	
UD12	Blank	
F10	901447-10	901447-10

VIC-20 Color (Pre-FCC Version)

Location	ROM #	Part Number
D5	901486-01	901486-01
D6	901486-06	901486-06
C7	901460-03	901460-03

VIC-20 Color (FCC Version)

Location	ROM #	Part Number
E11	901486-01	901486-01
E12	901486-06	901486-06
D7	901460-03	901460-03

D. O. S. 1.0 — 2040 Dual Disk Unit

Location	ROM #	Part Number
UL1	901468-06	901468-06
UK1	Blank	
UH1	901468-07	901468-07
UK3	901466-02	901466-02
UK6	901467	901467

D. O. S. 2.1 — 4040 Dual Disk Unit

Location	ROM #	Part Number
UL1	901468-12	901468-06
UK1	901468-11	901468-11
UH1	901468-13	901468-13
UK3	901466-04	901466-04
UK6	901467	901467

D. O. S. 2.5 — 8050 Dual Disk (Micropolis)

Location	ROM #	Part Number
UL1	901482-03	901482-03
UH1	901482-04	901482-04
UK3	901483-03	901483-03
UK6	901467	901467

**D. O. S. 2.5 —
8050 Dual Disk (Tandon)**

Location	ROM #	Part Number
UL1	901482-07	901484-07
UH1	901482-06	901482-06
UK3	901483-04	901483-04
UK6	901467	901467

**D. O. S. 2.6 —
2031 Single Disk Drive**

Location	ROM #	Part Number
U5F	90148x-xx	90148x-xx
U5H	90148x-xx	90148x-xx

2022 (Tractor Feed) Printer

Location	ROM #	Part Number
U11	901472-03	901472-03

2023 (Friction Feed) Printer

Location	ROM #	Part Number
Ull	901472-02	901472-02

2023 Printer — Interim Fix

Location	ROM #	Part Number
U11	901472-03	901472-03

2022 and 2023 Printers — Interim Fix

Location	ROM #	Part Number
U11	901472-04	901472-04

2022 and 2023 Printers — Final Fix

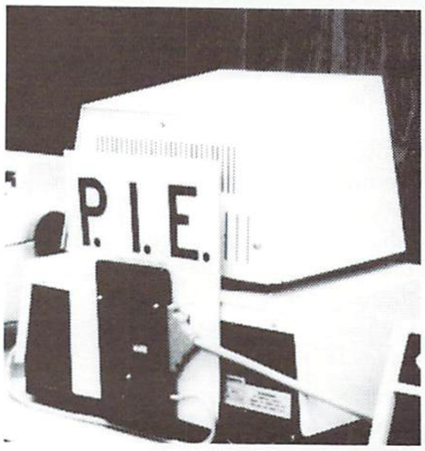
Location	ROM #	Part Number
U11	901472-07	901472-07

4022 (Tractor Feed) Printer

Location	ROM #	Part Number
U11	901472-04	901472-04 ■

PIE-C

PET/CBM* IEEE-488 TO PARALLEL PRINTERS By LemData Products



P.I.E.-C MEANS—Professional design, Indispensible features, Excellent quality and Cost effectiveness. You can't buy a better parallel interface for your PET/CBM.

Our P.I.E.-C will interface your PET/CBM through the IEEE-488 bus to . . . . . .

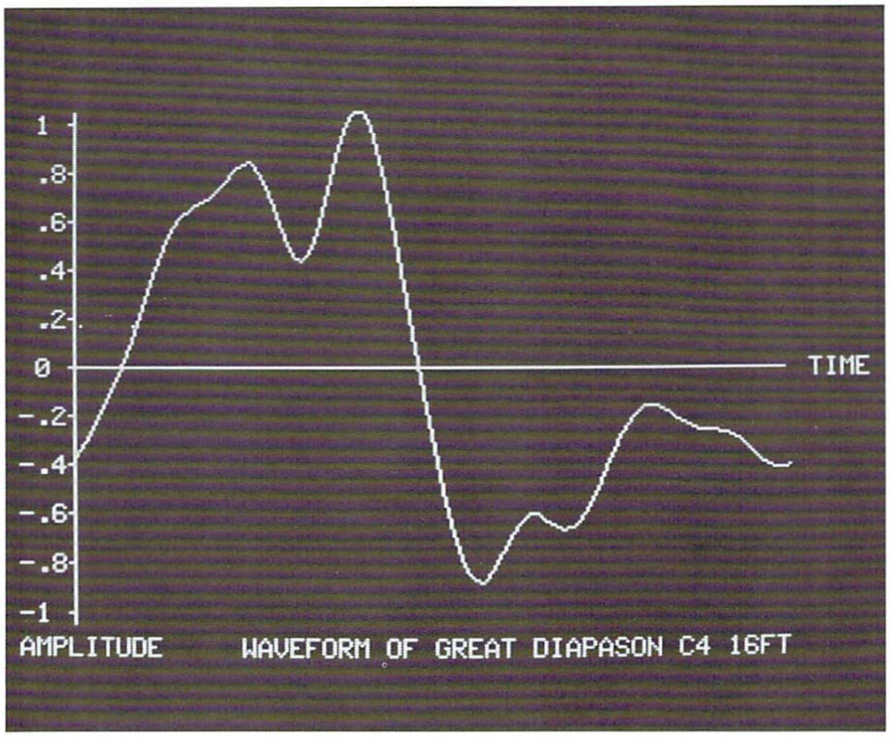
the NEC Spinwriter, the C. Itoh Starwriter, printers by Centronics, Epson, Anadex, Escon Products, the Paper Tigers by IDS, the MIPLOT by Watanabe, the DIP printers, the AJ-841, the OKIDATA printers, plus ALL OTHER parallel ASCII printers.

Assembled with custom case, CBM-TO-ASCII code converter and appropriate cable, the **P.I.E.-C** is only $129.95 (+$5 S&H). Md. Res. +5% tax. Specify printer and CBM models.

LemData Products, P.O. Box 1080, Columbia, Md. 21044 Phone (301) 730-3257
*PET/CBM are trademarks of Commodore Business Machines

"DEALER INQUIRIES WELCOME"

MTU Integrated Visible Memory Board & Keyboard Graphics Package



While many people tend to think of Commodore computers for their data processing capabilities in business, educational, or personal applications, few realize the power of graphics ǒn these systems and the myriad of ways high resolution graphics can be put to use.

First, let's define what graphic capabilities are available on standard Commodore systems. Each computer, from the VIC 20 to SuperPET contains the same PET Graphic Character set residing in ROM (Read Only Memory). This character set consists of 128 graphic characters and symbols, including various horizontal and vertical lines in different positions, corners, numerous shadings of different sizes, and special symbols such as check marks, hearts, etc.

Each computer has a "graphic" mode that selects the graphic character set, which is designed to give you much higher resolution than would be possible using straight text characters.

This higher resolution is possible because graphic characters occupy one-quarter of the "cell" or area that a full text character does. As such, if you were using a CBM 8000 or 9000 Series computer with 80 characters per line by 25 lines, there would be 8000 possible graphic locations rather than the 2000 locations available for text characters. All 8000 locations can't be used at the same time, but you can still plot denser graphics as compared to normal text mode.

However, in certain applications the resolution available from using graphic characters is not nearly enough and an alternate way to design graphics is needed.

Two Products from Micro Technology Unlimited (MTU) are uniquely suited to the needs of those desiring high resolution graphics on PET or CBM computers.

The MTU Integrated Visible Memory board is the hardware end of their system and the Keyword Graphics Package supplies sophisticated software for creating high resolution graphics.

MTU's K-1008-6 Visible Memory board contains 8K of RAM, 320 by 200 dot matrix display, a light pen register, ROM socket expansion and a KIM BUS interface. Versions are available for all configurations of PET and CBM computers and can be ordered by the version of BASIC you have (either 1.0, 2.0 or 4.0) and the memory size (16K or 32K).

The installation of the MTU board is very straightforward, but does require some care. Depending on which computer you own, the first step may be modifying a strip of jumpers on the computer's main CPU board. This is done by simply breaking a connection with an X-acto knife or making a new connection with a drop of solder. In many cases nothing needs to be changed.

The next step is to plug in a small printed circuit board onto the PET memory expansion connector. A power cable then runs from this small board to the rectifier diodes onthe PET's CPU board. These wires are easily attached with clip-on connectors.

A ROM from the main CPU board may also have to be reinserted in the Visible Memory board. This is due to a bug in the PET's address decoding logic.

From this point, a ribbon cable connects the small board (on the memory expansion bus) to the Integrated Visible Memory board. The last step is reconnecting the video connector from the computer's monitor to the small PC board. The MTU board is normally installed in the top part of the computer's enclosure with a special mounting kit. However, it can be left sitting on top of the CPU board, if you make sure the MTU board is insulated properly. Another option is leaving the board outside the cabinet entirely. But using the recommended mounting procedure is the best bet.

All components are of the highest quality and care has been taken to make the installation process as foolproof as possible. Connectors are keyed so cables are inserted properly. And the clear, step-by-step instructions take you through each procedure, with test points along the way. This ensures that each part is installed properly before going on to the next step.

The Visible Memory board actually performs a number of functions under software and jumper control. First, it contains 8K of RAM memory, useful for graphics and straight memory expansion. Next, there are five ROM sockets which can hold a variety of ROMs such as Tool Kits, protection ROMs, etc. A KIM expansion bus is also available for further expansion. Finally there is light pen option for the input of graphic data.

In operation, users can select from regular PET video, Visible Memory video, both signals overlayed, or blanking the screen without erasing it. This is accomplished by POKING a value into the video control register.

An enable control register lets you select which ROM on the MTU board is being addressed. This feature allows more than one ROM to share the same address space.

Programming the Visible Memory board to display text and graphics is easy. The display is essentially a matrix of dots with 200 rows of 320 dots per row. For addressing purposes, the dots can be numbered from 0 to 63,999, with dot 0 being the upper left-hand corner dot, dot 319 being at the upper right-hand corner. Eight horizontally adjacent dots make up one byte of memory with the position of the dots on the display corresponding to the position of the bits in the byte. Thus, dot 0 is the leftmost bit (bit 7, weight of 128) of the first byte in the Visible Memory. Conversely dot 319 would be the rightmost bit (bit 0) of the 14th byte.

Usually graphics programming is performed using the X-Y method of identifying a particular dot position. The dot position can be found using the simple equation:

```
DOT# = (199-Y) * 320+X.
```

Then, a byte address and bit number are calculated. Finally, the dot may be turned on with machine language instructions or their BASIC equivalents.

Although it is a lot of fun to build up graphic subroutines yourself, it is time consuming and many users prefer to get right down to using the Visible Memory board for applications.

The MTU Keyword Graphic Package (KGP) is a set of utility routines written in machine language but callable from BASIC statements. Besides high speed screen clear and point plotting, they also do point erase, line plotting given X and Y endpoints, and text plotting. Alternatively, text may be printed in the normal way on the screen with the video overlay option.

KGP extends the command repertoire of PET BASIC to include over 45 graphics commands especially for the Visible Memory Board.

For example, if a solid vector between coordinates 35,21 and 177,73 is desired the BASIC statement:

```
130 LINE 35,21,117,73
```

would draw the desired line.

A caption located at X=220 and Y=123 could be generated by simply coding:

```
710 MOVE 220,123
720 CHAR "MARKET INDEX"
```

In addition, the package provides some powerful advanced functions not found in other graphic software such as automatic coordinate transformation (both translation and scaling), solid or dotted lines, keeping track of up to 4 different display windows, subimage definition, and even a "scroll" command to facilitate the

→

CBM/PET? SEE SKYLES ... CBM/PET? SEE SKYLES... CBM/PET? SEE SKYLES ... CBM/PET? SEE SKYLES ... CBM/PET? SEE SKYLES ... CBM/PET? SEE

"Should we call it Command-O or Command-O-Pro?"

That's a problem because this popular ROM is called the Command-O-Pro in Europe. (Maybe Command-O smacks too much of the military.)

But whatever you call it, this 4K byte ROM will provide your CBM BASIC 4.0 (4016, 4032) and 8032 computers with 20 additional commands including 10 Toolkit program editing and debugging commands and 10 additional commands for screening, formatting and disc file manipulating. (And our manual writer dug up 39 additional commands in the course of doing a 78-page manual!)

The Command-O extends Commodore's 8032 advanced screen editing features to the ultimate. You can now SCROLL up and down, insert or delete entire lines, delete the characters to the left or right of the cursor, select TEXT or GRAPHICS modes or ring the 8032 bell. You can even redefine the window to adjust it by size and position on your screen. And you can define any key to equal a sequence of up to 90 key strokes.

The Command-O chip resides in hexadecimal address $9000, the rightmost empty socket in 4016 and 4032 or the rearmost in 8032. If there is a space conflict, we do have Socket-2-ME available at a very special price.

Skyles guarantees your satisfaction: if you are not absolutely happy with your new Command-O, return it to us within ten days for an immediate, full refund.

Command-O from Skyles Electric Works..**$75.00**

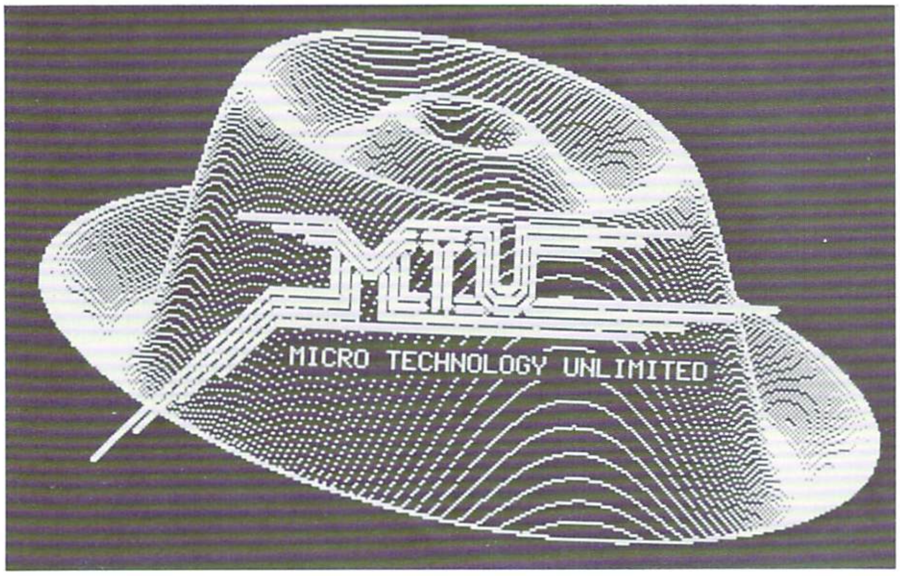
Complete with Socket-2-Me.. **95.00**

Shipping and Handling........................*(USA/Canada)* **$2.50** *(Europe/Asia)* **$10.00**

California residents must add 6%/6½% sales tax, as required.

Skyles Electric Works
231E South Whisman Road
Mountain View, California 94041
(415) 965-1735

Visa/Mastercard orders: call tollfree **(800) 227-9998** (except California). California orders: please call **(415) 965-1735.**



programming of animated displays. All four windows can scroll rapidly enough to appear as though they are scrolling simultaneously.

The most common use for graphics is plotting graphs and other images from mathematical functions or measured data. When KGP is initialized the origin of the plotting grid (0,0point) is set to the lower left corner of the screen. X coordinates in the range of 0 through 319 and Y coordinates in the range 0 through 199 are acceptable.

For greater flexibility, a coordinate offset can be specified which has the effect of moving the origin. Independent X and Y scale factors can also be specified which will shrink or expand the image in either or both dimensions.

A grid larger than the display screen is also available. If a large figure is drawn, different parts of the overall image may be seen by resetting the offset and redrawing the entire image.

Two different kinds of subimage capability are built-in. Subimages are useful for situations where shapes from a library are to appear in an image several times at different locations. Examples include various types of schematic diagrams and even ordinary text characters.

This subimage capability can be extended to provide animation from BASIC nearly as good as dedicated machine language programming. Entire images can be redrawn in different locations at high speed by merely specifying the new location of the subimage. The "scroll" function will move entire portions of the screen from one location to another without the need to erase and redraw the image in the new location.

The real power of KGP resides in the extended BASIC commands that actually perform the graphics. Without going into each in detail, a few will be covered to give you a feel for their operation.

Four video selection comands switch between video displays. *VISMEM* displays the Visible Memory display. *PETMEM* restores the PET character display. *PVMEM* will cause both displays to be shown together and *NOMEM* blanks the screen.

Three commands are provided for selecting background and display colors. *NRMDSP* command selects a black background, which is considered normal. *RVDSP* selects a white background. *FLPDSPS* changes to the opposite background color. Two clear commands erase either the entire screen or clear rectangular portions of the screen, as specified.

Three plotting modes are available for maximum flexibility, which are selected by the *GMODE* command. In mode 1, all plotted points are set the opposite color of the background. Mode 0 is a special case. Here, all points plotted are made the opposite of their previous color rather than opposite the background color.

The KGP is capable of directly plotting points and straight lines. The location of points and lines on the screen are defined by the X and Y coordinates. In many of the plotting commands the location of a "drawing cursor" is important. The *MOVE* command followed by X,Y coordinates is used to set the position of this cursor.

To plot a point at the current location of the drawing cursor, simply enter the command WRPIX. *This writes a pixel according to the current graphic mode.*

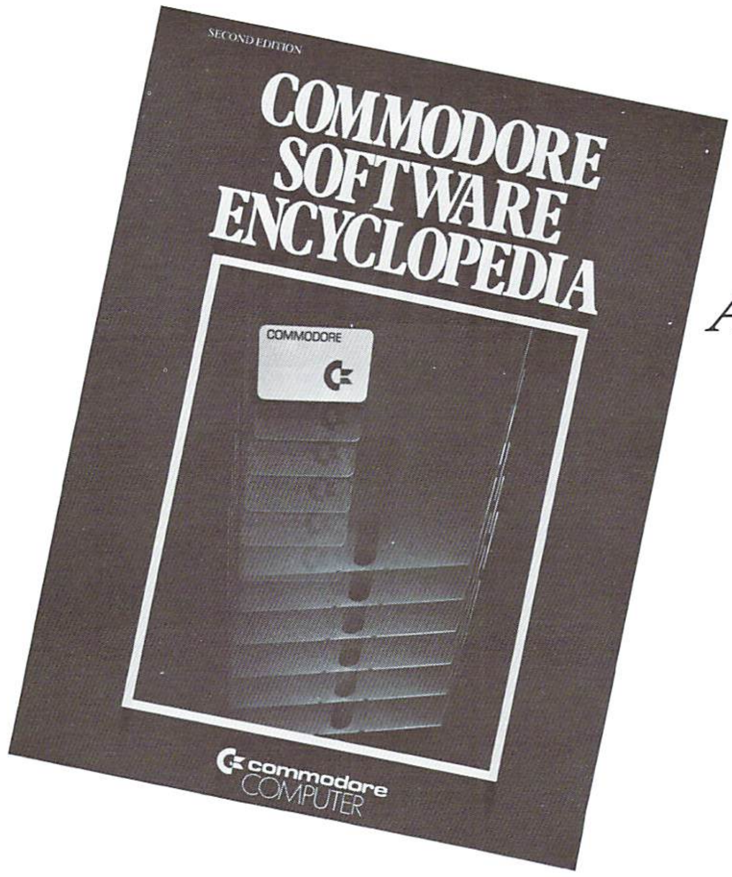
Two commands are provided for plotting lines between points. The LINE command will plot a line between any pair of endpoints without having to set the drawing cursor first. The *DRAW* command will draw a line from the drawing cursor position to an endpoint specified. This command is useful for drawing figures made of end-to-end connecting lines.

Also included within KGP are command to display a character string, set scaling and offset, rotate a figure, set display windows, cursor movement, create dotted lines, flip the display dots in certain areas, create shape tables, and a number of utility functions.

In all, the MTU Visible Memory board and Keyboard graphic package for PETs and CBMs is the most comprehensive high resolution system available for Commodore systems. The hardware is complete and easy to install, and, combined with the KGP software, you can easily create sophisticasted high resolution graphics that will rival larger systems. ■

—Mike Heck

WE WROTE THE BOOK ON SOFTWARE!



OVER 1,000 APPLICATIONS ONLY $9.95

The Commodore Software Encyclopedia is a comprehensive directory of over 1000 programs for business, education, personal, and recreational applications. Listings are provided for such categories as:

- Business
  including Dow Jones Portfolio Management, Legal Time Accounting, Accounts Receivable, Inventory Systems, Tax Preparation, Real Estate Programs, Payroll, General Accounting, Mailing List Management, and Visicalc.
- Word Processing
- Utilities
- Engineering Aids
- Personal Aids
- Games
- VIC-20
- Publications
- Software Vendors
- Hardware Overview
- Overseas Suppliers
- Education
  including Math, Science, Social Studies, Verbal Skills, and Administration.

Whether you're interested in personal computers or own a Commodore, this directory can be an invaluable one in locating just the right software package for you.

Pick one up at your local Commodore dealer today or return this coupon.

Please send ____________ Software Encyclopedia(s) @ $9.95 +2.00 postage and handling. Enclose check or money order.
Name ____________________
Address ____________________
____
____
____

Return to: Commodore Magazine
Commodore Business Machines
681 Moore Road
King of Prussia, Pa. 19406

☐ I own a Commodore Computer Model ________

commodore COMPUTER

The Friendly PET — Screen editing

One of the friendliest things about the PET, CBM and VIC is the way they allow you to make a change or correction. If the line on the screen is wrong—whether it is a program line or a direct command—we can move the cursor back and type over the line. Pressing the RETURN key will make the change take effect.

Correcting Programs.
This is very handy for programs. When your first program attempts result in a message such as ?SYNTAX ERROR IN 350 you can list 350 to see the problem. If line 350 happens to say PWINT X, you can move the cursor back, type R over the W to give PRINT X, and strike RETURN. The line has been corrected with a minimum of typing on your part.

If you need to make an insertion into your program, you may use the INSERT key. If the mistake was PINT X, the technique is to position the cursor over the I, hold down the SHIFT key, and press INST for insert; the computer will open up space and you can type in the missing R. On the other hand, if the error was PHRINT X you'll want to make a deletion: place the cursor over the R, press the DEL key to delete, and the H will disappear. In either case, don't forget to press RETURN to make the change permanent.

If you hapen to goof in making the change, start over. In this case don't press RETURN. Hold down SHIFT and then press RETURN: this will take you to the next line without any program change being made.

Shifted-RETURN is quite a handy key combination to know for many reasons. If you wanted to leave a note on the computer's screen for someone to read, you might type MARY - PUT THE CAT OUT. At this point, striking RETURN would cause the computer to try to "perform" the line, and you'd get ?SYNTAX ERROR. If you press

Shifted-RETURN, however, you'll just go to the next line and the computer won't try to do anything with the contents of the previous line.

The INSERT key has some special rules. After you have pressed the INSERT key a number of times (don't forget to hold down SHIFT) there will be an open space on the screen where you can insert the new characters. At this point, you'll be in "programmed cursor" mode. This means that the cursor keys don't move the cursor; instead they will print as special reversed characters. This is the same way that the PET behaves after you press the quote-mark key, with two important exceptions: the computer remains in this mode only for the number of characters to be inserted; and the Delete (DEL) and Insert (INST) keys work in a different way. More about this another time; in the meantime, you'll get used to them quite quickly.

A problem sometimes crops up if a program line is too long. Sometimes this means that there's no extra space available to make a desired insertion—eighty characters is the screen limit. Worse, the line is too long to start with; it occupies over 80 characters even before we make a change. It might be more sensible to change it to two lines and relieve the crowding; but if you must, the trick is to look through the line to find a keyword that can be abbreviated. PRINT is the most popular, since it can be rewritten as a question mark. Close up the space, making sure that everything is packed into the 80-column work area, and then make the change if it fits.

The Direct Approach.
Direct lines—Basic commands typed in without a line number so they are executed right away—are usually easy to fix. If you mistype LOAD "PROGRAM" so that it comes out LOUD "PROGRAM", don't be dismayed by the ?SYNTAX ERROR. You can slip the cursor back, change the U to an A, press RETURN and the load will take place.

Correcting mistakes in Direct lines can leave a cluttered screen. When I try to load BOTTLESHIPS from the disk, I get several lines which tell me that there's no such program. When I move the cursor back and correct to BATTLESHIPS, the following lines don't go away unless written over. It looks messy, but works OK.

There's a sharper problem when I ask a direct statement to print a number. If I ask the PET to calculate 4*5*6, yielding a product of 120, and then decide that I really want addition, I can go back and change the asterisks to plus signs. The PET will now produce a total of 15, but the last digit of the previous answer won't be wiped out; the Zero will be left on the screen and our sum will look like 150 instead of 15. The solution? Wipe out any numbers you want recalculated so that the new values will print on a fresh line.

Special screenings.
When you press RETURN, the PET sees only what's on the screen. You may have done deletions, insertions, and changes but the final screen result is all that counts. This is true of program lines, direct commands, and responses to program INPUT statements.

You may want to run a program several times while testing, with similar answers to INPUT questions on each run. With screen editing, it's a snap. After you have run once, move the cursor back to the RUN statement. Press RETURN (no need to type RUN: it's on the screen). For each INPUT, the cursor will appear over the answer you typed on the previous run. If you want to go with the same response this time, just press RETURN and the program will accept the same input from the screen. If you want to change, type your new input.

Here's a hint of advance techniques that you'll learn as you become more familiar with your computer. You can actually get the PET to type its own input—even its own program changes—to the screen. Then, with a stroke of the RETURN key, you can activate the input or program change. When used for INPUT activities, this provides a "default" input for the user. As a program changes, the program could suggest DATA statements that it would like to see included in a future run. Mind boggling! At this rate, the computer could program itself and make us all obsolete.

At least, the computer still needs us to press the RETURN key; it can't do that by itself. Or can it? Technical tyros suggest that POKE 158,1:POKE 623,13 (or on Original ROMs, POKE 525,1:POKE 527,13) would actually cause the PET to send a carriage return to itself. . .

First Programming Steps

The first programs that a beginner writes tend to be simple. That's good, of course: the programmer is developing skills which will be useful when he tackles more ambitious jobs. Here are a few suggestions on how to go about these early projects; the emphasis will be more on sound practices and clear style rather than clever coding methods. Some of the suggestions might be useful for experienced programmers, too.

Try to lay out your programs in "blocks". Each block should have a clear, simple function. One block might do an input job, another might calculate, and a third generate output. If you start planning a program by thinking out the blocks you will need, your program will be better planned. Some programmers make each block into a subroutine so that the main program simply calls in these units as needed.

Title each section or block with a remarks REM statement. You don't have to put comments on each line, but it's useful to be able to find a section of code quickly. Perhaps you think that you can remember the code—after all, you wrote it—but wait a couple of months. It's amazing how a crystal clear progam can suddenly become gibberish after you've been away from it for a while. Leave yourself some highway markers so that you can find your way around later.

Name your variables in a semi-meaningful way. Totals can start with the letter T, counts with a C, and so on. I'm not a fan of large alphabetic names, since they have pitfalls:

➔

TERRIFIC is a great label, but it doesn't work since the keyword IF is hidden in the middle. Can you find the hidden keywords in GRANDPA, CATNIP, CRUNCH and FRONT? It's fun to play word games, but not when you're trying to write a program. I prefer a single letter followed by a numeric: T4, B7 and so on. By the way, don't forget that variable B has nothing to do with integer variable B% or string B$ or for that matter array variable B(3). They are all completely independent values.

Don't let anyone hustle you about program size or speed. If others write in less memory and fewer milliseconds, let them. You'll have space enough for most of your programs and the tenth of a second saved in run time won't give you time for a cup of coffee. On the other hand, do look for better methods. Better isn't always faster or smaller, but you'll recognize it when you see it.

Keep track of your variables; it's useful to make a list on a sheet of paper. That way, you won't accidentally use variable X for two different jobs and get them mixed up. In fact, it doesn't hurt to do paperwork planning before turning your computer on. There's a kind of "heat" in working directly on the machine that sometimes leads to hasty programming. A little leisurely planning beforehand can generate sounder and better programs.

Don't be afraid to write loosely. The fanatic who tells you that you'll save memory and time by compacting FOR M = S TO P STEP V into FORM=STOPSTEPV is steering you wrong in most cases. If legibility costs you four bytes and one millisecond, take it: it's a bargain.

If your program doesn't work right the first time, don't lose heart. It happens to most of us. The easy errors are where the computer tells you where the problem is, most commonly ?SYNTAX ERROR IN . . . The problem will likely be obvious when you look at the line; if not, you can try rewriting it slightly to see what happens. The hard errors are where the computer doesn't stop, but gives you the wrong answers.

Debugging can be great fun if you can take the right attitude. Look at the variables: you can call them up with direct PRINT statements. Change them if it suits your purpose. Put STOP commands into your program and check everything out when you come to the halt. You can resume where you left off with CONT. Using the RUN/STOP key to break your program in mid-execution is less precise but will also do the job.

Getting a program together can be a rewarding experience—not necessarily rewarding in money, but in a sense of accomplishment. Each program will be a work of art, done in your own style. When you put your signature to your latest masterpiece, you'll feel good about it if you've used good coding craftmanship.

Half a dialogue — Inputting

Asking a porgram to go and get input from the user is a subtle thing for beginners. When you write your first programs, it's hard to look ahead and see the program independently communicating with the user. "If the program needs a value, I'll program it in right now . . ." It takes a level of sophistication to imagine a program accepting working values at a later time, when it runs, and using different values supplied by the user in different runs.

There are three fundamental ways of checking what the user is doing at the keyboard: INPUT, GET, and a PEEK. We'll talk about each, and its uses.

INPUT.
The INPUT statement does a lot of work for you. It's certainly one of the most powerful statements in BASIC. Some of us would like to see it more powerful, and some would like to see it less sophisticated; for the moment, we'll have to accept it as it is.

When you give the command INPUT in a program, a prompting question mark is printed and the cursor begins to flash. Your program is held in suspended animation; it will not resume operation until RETURN is pressed. There's no code which allows something like:

INPUT M:IF (NO REPLY IN 15 SECONDS) GOTO . . .

Your code will hang on the INPUT statement forever if the user doesn't reply.

When the user presses RETURN, INPUT takes the information from the screen. It doesn't matter if the user wandered back and forth, changing, deleting and inserting; INPUT looks only at the screen which is the result of his/her actions. In fact, if there's something on the screen that the user didn't type, INPUT will take that too. This can be useful for prompting: you can arrange to type a sample response on the screen, and the user will be able to press RETURN to have that response entered. As INPUT takes the information from the screen, it trims away all leading and trailing spaces; other than that it takes the whole line, even though it may not need it.

Now INPUT starts to plow through the line, digging out the information you need for your program. If it's looking for a number it will not like to find a string, and will ask, REDO FROM START. If it's looking for a string, it won't mind a number at all: it will accept it as a string.

Road signs for INPUT.
Whether INPUT is looking for a number or a string, it will stop its search when it finds one of three things; comma, colon, or end of line. If it finds a comma it will assume that more information will be needed later in the INPUT statement; if it finds a colon or end of line it assumes that there is no more useful input from the user. If it needs more, it will ask for it.

Suppose you need to input a string that contains a comma or a colon, such as ULYSSES M PHIPPS, PHD. or ATTENTION: JOHN, MARY. Since INPUT normally stops at the comma or colon character, we need to do something. The answer is easy: the user must put the desired input in quotes: "ATTENTION: JOHN, MARY" and the whole thing , commas, colon and all, will be received as a single string.

Keep in mind that the INPUT statement allows prompting. INPUT "YOUR NAME";N$ causes the computer to type YOUR NAME? and wait for input. That's a good human interface; helping the user along.

If a user presses RETURN without supplying any information on the screen, programs on the PET/CBM will stop. There are several ways to prevent this from happening; the easiest is to add a "canned reply" to the input prompt message. When you are writing the INPUT statement prompt (such as YOUR NAME) add two extra spaces and, say, an asterisk character; then type three Cursor-Lefts (they will print as an odd-looking reversed bar) and close the quotes on the prompt. Finish the INPUT statement in the usual way: a semicolon behind the prompt and then the name of the variable to be input. Now: the asterisk or whatever will print to the right of the prompt and question mark. Unless the user overtypes it, this character will be received from the screen as input—and the program won't stop.

One last comment: don't forget that INPUT can accept several values. You can say INPUT N$,A$,C$ and allow the user to type JOE BLOW, CITY HALL, DENVER. It's often better to use separate input statements: users can respond better when prompted for each piece of information.

GET and PEEK: a preview

GET isn't as clever as INPUT, but it has valuable uses. First of all, it doesn't wait; if a key isn't ready in the keyboard buffer, the GET statement lets BASIC continue. Secondly, keystrokes received with GET don't affect the screen unless you, the programmer, decide to allow them to do so. This means that you have much more control over what the user can do.

There's a PEEK location (PEEK(151) on most PET/CBMs, PEEK(515) on Original ROMs, and PEEK(197) on the VIC that tells you whether a key is being held down or not. This can be useful to avoid the situation where a user needs to press the same key repeatedly to cause some action; you can program so that the key repeats its action if it is held down.

We'll talk in more detail about the GET and PEEK next time around. They are more fun in some ways than the INPUT statement . . . but they call for quite a bit more programming work to be done.■

BRILEY SOFTWARE

Official Notice!!!

All PET/CBM owners who have not seen our line of useful programs are hereby requested to contact the nearest computer store right away! These programs are also described in the Commodore Software Encyclopedia. Don't delay. Check us out today!

If your nearest Commodore dealers have no information, please send their address to us. We will send both them and you a catalog promptly!

Briley Software is a software publishing firm dedicated to supporting PET/CBM micro-computers (both old and new ROMs).

P.O. BOX 2913
LIVERMORE, CA
94550-0291
(415) 455-9139

CBM/PET? SEE SKYLES ... CBM/PET?

SEE SKYLES... CBM/PET? SEE SKYLES

... CBM/PET? SEE SKYLES ... CBM/

PET? SEE SKYLES ... CBM/PET? SEE

"They laughed when I sat down at my PET and immediately programmed in machine language... just as easily as writing BASIC."

With the new Mikro, brought to you from England by Skyles Electric works, always searching the world for new products for PET/CBM owners. A 4K machine language assembler ROM that plugs into your main board. At just $80.00 for the Mikro chip, it does all the machine language work for you; all you have to do is start laying down the code.

The Mikro retains all the great screen editing features of the PET...even all the Toolkit commands. (If you own a Toolkit, of course.) Sit down and write your own machine language subroutine. The program you write is the source code you can save. And the machine language monitor saves the object code. The perfect machine language answer for most PET owners and for most applications. (Not as professional as the Skyles Macro-TeA...not as expensive, either.)

A great learning experience for those new to machine language programming but who want to master it easily. Twelve-page manual included but we also recommend the book, "6502 Assembler Language Programming," by Lance A. Leventhal at $17.00 direct from Skyles.

Skyles guarantees your satisfaction: if you are not absolutely happy with your new Mikro, return it to us within ten days for an immediate, full refund.

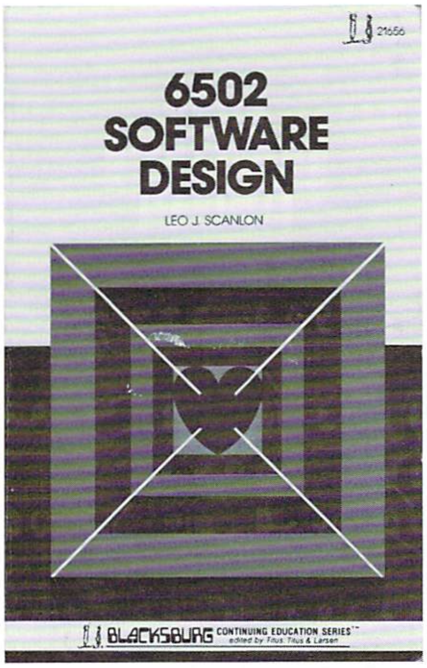
Skyles Mikro Machine language assembler......**$80.00**
"6502 Assembler Language Programming" by Leventhal...... **17.00**
Shipping and Handling......*(USA/Canada)* **$2.50** *(Europe/Asia)* **$10.00**

California residents must add 6%/6½% sales tax, as required.

Skyles Electric Works
231E South Whisman Road
Mountain View, California 94041
(415) 965-1735

Visa/Mastercard orders: call tollfree **(800) 227-9998** (except California). California orders: please call **(415) 965-1735.**

BOOK REVIEW



6502 Software Design

by
Leo J. Scanion

ATTENTION all proficient CBM/ PET BASIC programmers who think the time is right to go forth into the 'mysterious' realm of assembly language. Yes, there really is a good introductory text (with examples) for utilizing the power of the 6502. Unfortunately it carries the not too straightforward title "6502 SOFTWARE DESIGN."

Although the AIM 65, Rockwell's 6502-based microcomputer has been used to generate and test the examples in the book, most of the programs (and all of the concepts) are valuable to CBM/PET users.

The book starts off with a brief, but informative history of S bit microprocessors, basically as a geneology of the 6502, including the actual circumstances of its design. The bulk of chapter one is a concise introduction to the 6502, its theory, design, and how it works.

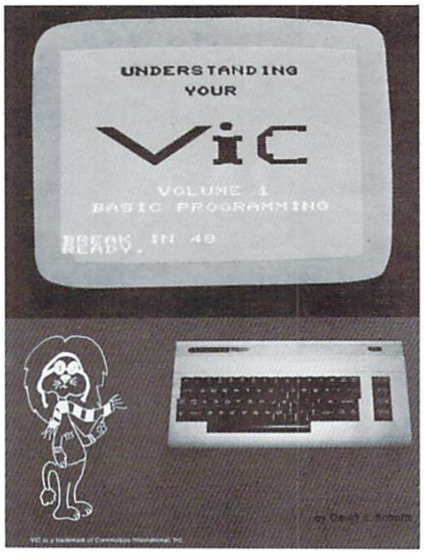
Chapters 2 and 3 include a rather complete, organized, well explained introduction to the 6502 instruction set and the powerful but confusing addressing modes of this processor, including numerous simple subroutines.

Chapters 4 through 6 advance the reader through the techniques needed to process and sort lists of data and create look-up and jump tables, as well as perform mathematical operations and convert data.

The final chapters 7 through 9 are concerned with interrupts, rests, and ways to transfer information between microprocessor and i/o devices (including the 6522 VIA & the 6520 PIA), and ultimately interfacing to the real world.

All in all, the book is quite informative. If, like me, you prefer real live examples to pounds of text, this may be just the book you need to get you into the wonderful world of machine language.

—John Stockman



Understanding Your VIC, Volume 1: Basic Programming

by David E. Schultz
Total Information Services
140 pages
(paperback with program cassette)

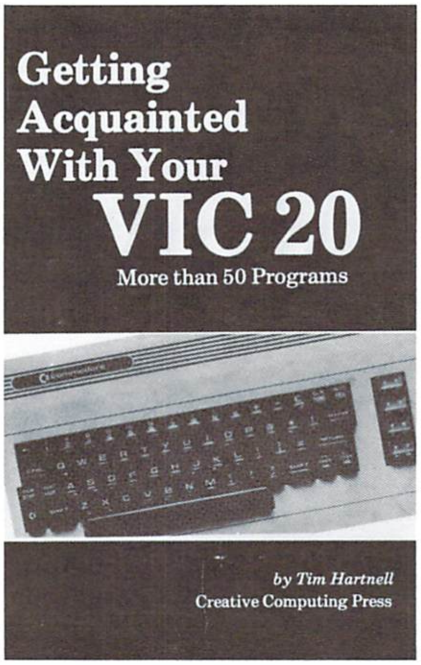
This book is a hands-on workbook on introductory programming techniques—at least in some parts. There's a 50-page section in the middle of the book that is a radical departure in style. However, once you get used to the conversational "you/VIC" format, it is easy to understand the book's approach to teaching BASIC.

The intent of this workbook seems to be the presentation of good introductory programming techniques. It illustrates, through examples, the functions and limits of each BASIC command. It covers all the commands and fuctions and even touches upon variables, plus supports them with good examples.

The three middle chapters attempt to describe abstract concepts of programming, but, unfortunately, things get *too* abstract. This information should be revised for the next edition. These middle chapters are followed by some good introductory material on number systems and computer logic.

Much of this workbook contains material updated from TIS workbooks for the PET/CBM Computers. Since most of the material in the beginning of the book is the same for the PET or VIC, you might be better off getting the PET edition instead, along with the TIS workbook on PET graphics, which has been a popular item for a few years now.

—Neil Harris



Getting Acquainted with Your VIC 20, More than 50 Programs

by Tim Hartnell
Creative Computing Press
132 pages (paperback), $8.95

This book is written for the novice programmer, someone who has just barely outgrown the VIC Owner's Manual. It is full of little programs for your amusement. The programs are annotated to help you understand how they work. Most of the programs are games, with random numbers getting a heavy share of the explanation.

The games are a little more sophisticated than those in the user manual, but nothing like the programs you can buy commercially on tape. Most of the casino games are considerably trimmed from their original forms, especially the Las Vegas simulations.

The book looks like it was originally written for another computer, since some of the commands in the programs do not apply to the VIC 20 and—even worse—do not work. For example, on page 16 of the book, line 165 of the program reads CLS, which is another computer's command to clear the screen. Some programs were rewritten for this book to conform to the VIC's screen size, but some were not. And the RETURN key is sometimes called the ENTER key, and once it is even called NEWLINE.

Just for the record, appendices C and D were reproduced from Commodore's VIC User Guide.

Overall, this book receives a lukewarm rating. Its main good point is that it tries to provide education in program techniques along with games. ■

—Neil Harris

PROJECTIONS & REFLECTIONS

COMDEX '81 marked another milestone in Commodore's advance towards market prominence. The exhibition, which ran from November 19th thru the 22nd, in Las Vegas, was attended by about eighty thousand people. If one could characterize the typical reactions to our booth, they would run from mild interest to extreme excitement.

If there was any message that was made clear by our presence, it was that the CBM product line is not only the price-performance leader, but the machine selected for software development by the widest range of professional software houses. The booth featured more than 50 demonstrations of hardware, peripherals, software and applications. Booth personnel included over two dozen outside vendors (some from overseas), as well as our own staff.

Just some of the more exciting stops around our booth included a communications device that supports IBM protocalls, by Davidson-Richards from England which is being marketed by Computer Marketing Services of Cherry Hill, NJ, a full-featured CBM basic compiler called PETspeed by Oxford Computer Systems from England, our large wordprocessing station showing Wordcraft 80, a self-contained battery backup for the CBM by ETC of North Carolina, and our own VIC and hard disk drive displays. Outside of our booth the Commodore product line was represented by companies that produced software as well as add-on firmware and hardware options.

For those of you who visited us in Las Vegas, I'm sure you will agree the turnout and support for our products was very gratifying, and for those of you that were unable to attend, you missed a good one!

In summary, I would just like to extend my personal thanks as well as Commodore's to all of the people that helped make the show a success, especially the outside vendors that came and represented their products in our booth. With all of the excitement your presence generated I'm confident that next year the Commodore booth will have to be twice as large to accomodate all of the products and representatives. ■

Merry Christmas
and a Happy New Year to you all.

—Paul Goheen
Software Product Manager

COMMODORE RETAIL PRICE LIST

Effective September 1, 1981

Product	Description		Retail
COMMODORE PERSONAL COMPUTER RANGE — VIC Series			
VIC 20	Full-Featured, expandable color computer system. 5K RAM. Including RF Modulator & TV switch box.		$ 299.95
C2N DATASETTE RECORDER	Cassette storage for PET/CBM/VIC		75.00
VIC 1515 GRAPHIC PRINTER	VIC dot-matrix printer, 30 CPS, 8" paper; prints full VIC character set; tractor feed.		395.00
COMMODORE EDUCATIONAL/SCIENTIFIC COMPUTER RANGE — 4000 Series			
PET®	System with graphic keyboard and numeric keypad. 12" display/40 characters.	4016N 4032N	995.00 1295.00
COMMODORE BUSINESS COMPUTER RANGE — 8000 Series			
CBM™ 8032B	Typewriter-style keyboard, numeric keypad, 80 column x 25 line display. 32K RAM, BASIC 4.0.		1495.00
CBM 64K MEMORY EXPANSION BOARD	64K Add On-Memory. Expands CBM 8032 to 96K RAM.		500.00
COMMODORE ADVANCED COMPUTER RANGE — 9000 Series			
SuperPET COMPUTER SP9000	Enhanced 8032 with additional 6809 microprocessor, total 134K memory including 96K RAM includes Waterloo microBASIC, Waterloo microAPL, Waterloo microPascal, Waterloo microFORTRAN and Waterloo 6809 Assembler interpreters.		1995.00
COMMODORE PERIPHERALS			
DISK DRIVES			
CBM 2031 SINGLE DISK	Single drive intelligent 5¼" floppy disk system. 170K (DOS 2.6)		695.00
CBM 4040 DUAL DISK	Dual drive intelligent 5¼" mini-floppy disk system. 340K (DOS 2.1)		1295.00
CBM 8050 DUAL DISK	Dual drive intelligent 5¼" mini-floppy disk system. 1 Megabyte (DOS 2.5)		1795.00
PRINTERS			
CBM 4022 PRINTER	80 Column printer with tractor feed Prints full PET graphics, variable line spacing, and programmable characters.		795.00
CBM 8023P PRINTER	136 Column, Dot-Matrix 150 CPS, bi-directional, graphics		995.00
CBM 8300P PRINTER	Letter quality, daisy wheel printer, 40 CPS, IEEE interface		2250.00
ADDITIONAL PERIPHERALS			
CBM 8010 MODEM	High performance 300 BAUD IEEE interfaced modem.		280.00
CBM 4010 VOICE SYNTHESIZER	Features phoneme syntheses for vocabulary construction. User port interface.		395.00

Product	Description	Retail

CABLES

IEEE-IEEE	Designed to connect more than one peripheral to any PET/CBM computer.	$ 49.95
PET-IEEE	Use for connecting one peripheral to any PET/CBM computer.	39.95

PET/CBM SOFTWARE

For a comprehensive list of all software available for PET/CBM computers, please refer to the Commodore Software Encyclopedia.

SOFTWARE ENCYCLOPEDIA	4.95

SOME SUGGESTED SYSTEMS

Personal System

VIC 20 Computer	$ 299.95
C2N Datasette Recorder	75.00
VIC 1515 Graphic Printer	395.00
	769.95

Educational System

CBM 4016 Computer	995.00
CBM 2031 Single Disk Drive	695.00
CBM 4022 Dot-Matrix Printer	795.00
	2485.00

Economy Business/Word Processing System (4000 Series)

CBM 4032 Computer	1295.00
CBM 4040 Dual Disk Drive	1295.00
CBM 4022 Dot-Matrix Printer	795.00
	3385.00

Advanced Business System (8000 Series)

CBM 8032 Computer	1495.00
CBM 8050 Dual Disk Drive	1795.00
CBM 8023P Dot-Matrix Printer	995.00
	4285.00

Advanced Word Processing/Business System

CBM 8032 Computer	1495.00
CBM 4040 Dual Disk Drive	1295.00
CBM 8300P Letter Quality Printer	2250.00
	5040.00

USERS NEWSLETTER

COMMODORE — THE MICROCOMPUTER MAGAZINE - 6 issues.	$15.00/yr.

This price list supersedes all previous Retail Price Lists.
Prices subject to change without notice.

commodore COMPUTER

681 Moore Road, King of Prussia, PA 19406 (215) 337-7100

LC00001

FILL OUT AND MAIL TODAY

THE MICROCOMPUTER MAGAZINE

commodore

Name____

Address________________________ Type of Business ______________

City________________ State ______________ Zip __________

Renewal Subscription______________ New Subscription______________

My subscription Began With Issue Number______________

Enclosed is a check or money order for $ ______________ ($15.00 for six issues)
(Canadian and foreign $25.00)

for______________ issues

Make Check payable to: **COMMODORE BUSINESS MACHINES, INC.**
681 Moore Road, King of Prussia, PA 19406
Attn: Editor, Commodore Magazine

COMMODORE BUSINESS MACHINES, INC. PRODUCT SPECIFICATIONS

VIC 20
$299.95*

Screen size: 22 char. x 23 lines
Memory: 5K expandable to 32K
User supplied color monitor or TV
4 internal amplifiers including:
- 5 octaves total range
- 3-tone (music) generators
- 1 sound effects generator

Peripherals include: cassette recorder, printer, modem, disk drive, and IEEE-488 interface, RS-232 interface, memory expansion module, 3K-8K-16K expansion cartridges
Features:
- Programmable function keys
- Standard PET BASIC — upwardly compatible to other CBM computers
- Full-size typewriter keyboard
- Graphics character set, upper and lower case letters
- Plug-in program/memory cartridges
- Low-priced peripherals
- Joystick/paddles/lightpen
- Self-teaching materials
- 16 colors
- High resolution graphics (176 x 184 dots)
- Programmable characters
- Full screen editing

*Includes modulator

PET®
$995.00

Screen size: 40 char. x 25 lines
Memory: 16K to 32K
IEEE-488 bus for disk, printer, modem and other intelligent peripherals
Eight-bit parallel user port with "handshake" lines
Supports Commodore C2N cassette and disk unit
Features:
- 128 ASCII plus 128 graphic characters
- 74-key professional keyboard
- Shift key gives 64 graphic characters
- Separate calculator/numeric pad
- Full screen editing capabilities
- 18K of ROM contains BASIC (version 4.0) with 9-digit floating binary arithmetic

Grants:
- 3 for 2 offer available for educational use.

CBM™ 8032
(Business Machine)
$1495.00

Screen size: 80 char. x 25 lines
Memory: 32K expandable to 96K
IEEE-488 bus for disk, printer, modem and other intelligent peripherals
Business software includes:
- Word processing
- VISICALC
- Inventory control
- Accounts receivable
- Accounts payable
- Payroll
- Dow Jones Portfolio Management
- Personnel files

Features:
- 128 ASCII, plus 128 graphics characters
- 73-key professional keyboard
- Separate calculator/numeric pad
- Full screen editing capabilities
- 18K of ROM contains BASIC (version 4.0) with direct (interactive) and program modes
- 9-digit floating binary arithmetic

Grants:
- 3 for 2 offer available for educational use.

SuperPET Computer
$1995.00

Screen size: 80 char. x 25 lines
Memory: 96K RAM (64K bank switched) IEEE-488 bus for disk, printer, modem and other intelligent peripherals High-speed RS232-C interface for additional compatibility with printer and modem
Communicates to Mainframes*
Compatible with 8032 software
2 microprocessors - 6502 and a 6809
Software includes: language interpreters, editor, operating system (supervisor), and an assembly language development system
- BASIC 4.0
- Waterloo MicroBASIC
- Waterloo MicroPascal
- Waterloo MicroFORTRAN
- Waterloo MicroAPL
- 6809 Assembler Development System, Linker/Loader

Features:
- 128 ASCII, plus 128 graphic characters, IBM/ACM, APL character set
- 73-key professional keyboard
- Separate calculator/number keyboard (line/text editor keypad usage in 6809 mode)
- Screen editing capabilities
- 38K of ROM contains BASIC (version 4.0) with direct (interactive) and program modes and 9-digit floating binary arithmetic

*Program file transfer between suitably equipped Mainframe. (Program upload/download between M/F and SuperPET). Programs will execute without modification.

commodore COMPUTER

Commodore Business Machines, Inc., Computer Systems Division, 681 Moore Rd., King of Prussia, Pa. 19406 (215) 337-7100

Commodore Business Machines, Inc.
681 Moore Road
King of Prussia, Pa. 19406

address correction requested

BULK RATE
U. S. Postage Paid
PERMIT No. 954
Philadelphia, Pa.