$2.95
U.S. FUNDS

JANUARY 1983 VOL. 1 ISSUE 2

COMMANDER

The Monthly Journal for Commodore Computer Users

Inside:

Game~Contest

Enterprise

Ravings of a Madman

0 74470 12426 01

Something new on the horizon.

Snow 'em with your VIC!

NEW

Simplify your printer set-up with

SMART ASCII $59.95

At last! A simple, convenient, low-cost printer interface. **It's ASCII:** connects the VIC or '64 to your favorite parallel printer (Epson, Microline, Smith-Corona TP-1, etc.). **It's SMART:** *translates* unprintable cursor commands and control characters for more readable LISTings.

Converts user port into parallel port with Centronics protocol, addressable as Device 4 or 5. **Three print modes:** CBM ASCII (all CAPS for LISTing); true ASCII (UPPER/lower case for text); and TRANSLATE (prints (CLR), (RED), (RVS), etc.). For **any size VIC** or the **'64**. Complete with printer cable and instructions.

UN-WORD PROCESSOR 2 . . . $19.95

The improved UN-WORD retains the practicality and economy of the original. Easy-to-use text entry and screen editing. Use with **any size VIC** (5K to 32K). Supports VIC printers, RS-232 printers, and now **parallel printers***, too. Handy user Menu selects: single- or double-space, form feed, print width, number of copies. Supports printer control codes. With complete documentation.

*Parallel printers require an interface. See SMART ASCII.

BANNER/HEADLINER $14.95

Make GIANT banners on your printer. Prints large characters across the page or sideways down the paper roll. . . how about a 10-ft. long "Welcome Home!". VIC or RS-232 printers.

VIC-20 is a trademark of Commodore Business Machines

VIC-PICs . . IMPROVED! . . $19.95

Now with **hi-rez draw routine** for your joystick *PLUS* hi-rez **dump to VIC printer.** Features 19 fascinating hi-rez *digitized* pictures. Capture *your* creativity, or ours, on paper. Amazing fun!

GRAFIX DESIGNER $14.95

Design *your own* graphic characters! Recall, erase, edit, copy, rotate . . . save to tape or disk for use in your own programs. Simple to use. Includes examples and demo routines.

GRAFIX MENAGERIE $14.95

Three-program set shows off VIC graphics potential for art, science, music, business . . . learn by seeing and doing. Contains BASIC **plotting routines** you may extract and use.

TERMINAL-40 . . . . . $29.95

Join the world of telecommunications in style: **40-character lines** and **smooth scrolling** text for easy reading! All software — no expensive hardware to buy. **4K** (or larger) **Receive Buffer** with optional dump to VIC printer. Function key access to frequently-used modes. Fully programmable Baud, Duplex, Parity, Wordsize, Stopbit, and Linefeed; supports control characters. Requires VIC-20, 8K (or larger) memory expansion and suitable modem. With 24 p. manual and Bulletin Board directory.

Introducing ***SOFTWARE FOR THE NEW COMMODORE 64***

'64 TERMINAL ($29.95). Same impressive features as TERMINAL-40: **smooth-scrolling,** 40-character lines, VIC printer dump, etc. **GIANT 24K Receive Buffer.** No memory expansion required; requires '64 and modem.

'64 GRAFIX SAMPLER ($19.95). Indulge in the graphics splendor of the '64. Interact with demos of techniques such as **plotting** of points, lines and 3-D objects; **drawing** in the hi-rez mode (joystick control); animating **sprites;** plus assorted graphics displays. Routines may be extracted for use in your own programs.

'64 PANORAMA ($19.95). Explore picture graphics on the amazing '64! Nineteen fascinating *digitized* pictures PLUS **hi-rez draw routine** for your joystick AND hi-rez **dump to VIC printer**. Capture our pics or *your* creativity on paper.

'64 BANNER/HEADLINER ($19.95). Make **GIANT banners and posters** with your '64 and printer. Supports VIC printers, RS-232 printers (requires interface), and parallel printers (requires Smart Ascii).

ORDER DESK
Open 9 am - 4 pm
(816) 254-9600
VISA/Mastercard add 3%
C.O.D. add $3.50

MAIL ORDER: Add $1.25 shipping and handling. Send money order for fastest delivery. VISA/Mastercard send #and exp. date (3% added). Missouri residents include 4.6% sales tax. Foreign orders payable U.S.$, U.S. Bank ONLY; add $5 shipping/handling.

All programs on high quality digital cassette tape.

Write for free brochure. Dealer inquiries invited.

MIDWEST MICRO associates
PO BOX 6148, KANSAS CITY, MO 64110

Table of Contents

STAFF

Publisher
THOMAS L. ROSENBAUM

Editor
ALICIA A. LINDEN

Assistant Editor
TERILYN M. FLOYD

Assistant Editor
LINDA L. LINDEN

Typesetting
GRANGE PRINTING

Consultant
EDWIN SUND

Printed By
GRANGE PRINTING

COMMANDER is published monthly by:
MICRO SYSTEMS SPECIALTIES, P.O. Box 98827,
Tacoma, Washington 98498

Subscription Rates	*Per Year*
U.S.	$22.00
Canadian, Mexican	$26.00
Surface Rates	$37.00
Air Mail	$54.00

For back issues, subscriptions, change of address or other information, write to:
COMMANDER
P.O. Box 98827
Tacoma, Washington 98498
(206) 565-6816

Call for Clubs and Newsletters Directory

To be included in the first edition of the Commander Clubs and Newsletters Directory, your club or publication must supply the following information:

1. *name of organization or publication*
2. *mailing address*
3. *contact person and telephone number*
4. *name of newsletter or publication*
5. *special interests*

Send your information to Clubs and Newsletters Directory, Commander, P.O. Box 98827, Tacoma, Washington 98498.

Commander - The Monthly Journal for Commodore Computer Users is published monthly by Micro Systems Specialties, P.O. Box 98827, Tacoma, WA. 98498. Domestic Subscriptions, 12 issues, $22.00. Second Class Postage pending at Tacoma, WA 98404 and additional mailing offices. Postmaster:Send address changes to: **Commander - The Monthly Journal for Commodore Computer Users, P.O. Box 98827, Tacoma, WA 98498.**

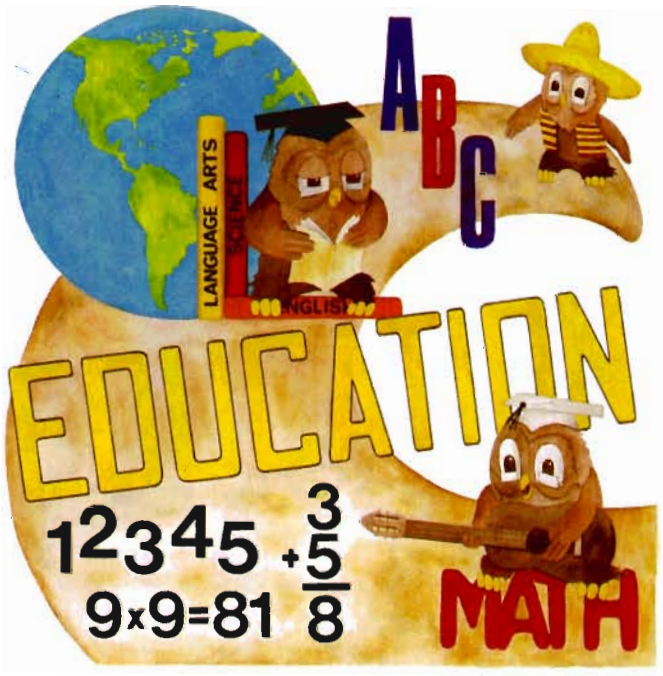
COMM*DATA SOFTWARE

GUIDES YOU AND YOUR VIC 20® DOWN ROADS OF ADVENTURE WITH:

- Maelstrom*
- Escape MCP*
- Gator Chase*
- Astro Command
- Caves of Annod
- Capture the Beast
- Whirlwind Rescue*
- Street Maze
- The Market
- Chivalry

THROUGH TRAILS OF CREATIVITY WITH:

- Sketch and Paint



ALONG THE PATH TO KNOWLEDGE WITH:

- Wordspot
- Math Tutor Series
- Alphabet Tutor
- Conversion
- Gotcha Math
- English Invaders
- Math Invaders Series

ASK FOR COMM*DATA COMPUTER HOUSE SOFTWARE AT YOUR LOCAL DEALER.

Or Send for FREE Catalog:

COMM*DATA COMPUTER HOUSE
320 Summit Avenue
Milford, Michigan 48042
(313) 685-0113

Dealer Inquiries Welcome.

Quality software also available
for Pet and Commodore 64 computers

MasterCard

VIC 20 is a Registered Trademark of Commodore Business Machines, Inc.
*High Res Full Machine Code Arcade Style Games.

VISA

Letters to the Editor

Well . . . My opinions don't change much, but there are a few new things available for me to form an opinion about so, for what it's worth, here is my opinion of several VIC games that we have recently acquired and, boy, do I have an opinion of advertisers that can't deliver on advertised products.

COMMODORE has recently offered "PINBALL SPECTACULAR" as an addition to their cartridge list. In my opinion it is the best yet. It utilizes **paddles** that operate smoothly, has excellent color and graphics and really great action. The action starts slowly enough that even I can keep up with it but it soon speeds up to the point where I just give up and let the kids play. The game combines some of the features of "BREAKOUT" with some new features to make for a lot of fun, and it can be played by two people—another improvement over most of the rest of the field.

We tried a couple of adventure games from AARDVARK TECHNICAL SERVICES. "QUEST" which is the first adventure that we've seen for the VIC that has graphics. The graphics are not great and the game is very difficult because there is so much about it that is random but it does have graphics. The object of the game is to acquire enough men and arms to storm the castle of the evil ruler and thus win the freedom of the kingdom.

The other AARDVARK game we tried was "PYRAMID," and it too is very difficult. We had to get some clues from a friend that had already solved the puzzle. The object of this game is to acquire some "treasures" and take them to a certain location. AARDVARK provides a listing with their games, which require memory expansion, so it is possible to get a lot of information from the listing if you get really stumped. We also encountered a couple of problems with ours, like the treasures would not all register in "PYRAMID" and some of the information on the screen in "QUEST" flashed on and off too fast to read. We have contacted AARDVARK about these problems and they are going to fix them. Both games are very difficult but fairly enjoyable. They do lack some features that Scott Adams enthusiasts like, like the save feature in the middle of the game so you don't have to start over every time.

My very low, low opinion this time goes to advertisers that cannot deliver on advertised products. I realize that there is a long lead time for most magazine ads so a company may not have an item that they have advertised by the time the magazine is printed. I'll accept reasonable delay while stock is being replenished but I get really incensed when a company back-orders an item on me in October, and keeps my money, and yet still cannot give me a delivery date in December. I finally had to call and cancel my order for a 16K expansion and ask for a refund so I could try to find one somewhere else.

I also got very excited when we located a 16K locally and they told us they would hold it for us for a couple of days until payday. When we went in two days later they said they didn't have it but would be getting one in in a "very few days." The upshot is that I still do not have my memory expansion, nor do I have my refund—such is life I suppose, but it would be nice if the first company had sent my money back after a week or so when they found they couldn't deliver.

If and when I get the 16K expansion I am going to try a new game that I received from PRICKLY PEAR SOFTWARE called "VIKING." Sounds really promising, for one to four players, adventure type and best of all, it is on DISK. I may have an opinion of it and some other things later.

Fred S. Dart
Salem, Utah

Dear Editor,

I think your new magazine is just great—especially for beginners like me. I just bought a VIC-20 a couple of months ago and need lots of support.

One application program I have been looking for is one to handle geneology. I have seen several for the Apple or TRS-80 but nothing for VIC. I would even like just a listing to type in myself and make improvements on. Perhaps some of your other readers might have a geneology application available. I would also like to see articles on how to design and write date bases with capabilities of sorting and relating items and variables.

Sincerely,
Larry S. Kramm
Antioch, CA

Dear Editor,

Just read your new magazine. Love it! How about this for a suggestion: A regular column called "Wish List" where readers could write in what they "wish" was available in soft- or firmware. Maybe suppliers would read it and come through. I'll start: I wish there was a plug-in key pad (numeric) for the 64.

Keep up the good work!

Jerry Chinn
Pasadena, CA

Editorial

The 1982 Christmas season has brought some profound changes to the personal/home computer market. The industry has seen constantly increasing growth through the worst economic times since the Great Depression and it seemed as it it would never end. In December, however, Wall Street surprised a lot of people by bringing the curtain down on many of the home computer stocks. The biggest loser, Atari, lost 25% of it's stock's value in one day and suffered a $1 billion dollar equity loss in one week. There were many contributing factors to this precipitous drop but one of the chief causes had to be the huge number of dealer order cancellations which were triggered by the dealers' inability to compete with the large volume discounts which the manufacturers were granting to the mass merchandisers such as Sears.

Most personal computer manufacturers, including Commodore, have been severely hampered by the lack of an effective retail distribution network. To develop such a network from the ground up would be very expensive and take a long time, neither of which is acceptable to a company on the fast track to success. The easily available alternative is to allow small retail electronics audio and/or computer stores to sell the computer. All that must be done is to establish a network of distributors to service the retail outlets. This strategy worked well in the infancy of the personal computer industry, however, as competition grew and computer prices and profit margins fell, the manufacturers began scrambling for alternative methods of selling more computers. The obvious solution was to turn to the mass merchandisers such as Sears, Penneys, K-Mart, etc.

The mass merchandisers can certainly sell a lot of computers but have you ever been able to have a saleman at one of these stores answer a technical question? Until the mass merchers train some personnel to help the consumer in this area, their niche in the distribution chain is uncertain. The mass merchers can discount computer prices to such an extent that the small retail stores cannot buy the machine at wholesale prices for less than the mass merchers offer it on sale. As such, the retail stores cannot compete with the mass merchandisers, but they can provide much better customer support.

Customer support, especially service, is the real key to a successful personal computer retail organization. The mass merchers are not yet equipped to provide timely service and show no inclination to do so, but they can offer the best price. Commodore's solution to this quandry has been to give the VIC-20 (because of its price) to the mass merchers and the 64 to the retail stores because much of the success of the 64 will depend upon customer service due to its much higher price.

How well personal computer manufacturers manage this dichotomy of their sales operation will dictate their overall success. Commodore's VIC-20 has attained volume sales leadership due to a successful advertising campaign and mass merchandising. The future of the 64 is as yet uncertain as the VIC-20—the product is well built and deserves to be tops.

News Releases

I publish this listing solely to promote the VIC 20 industry. By doing this, I hope to improve the level of technical support the VIC receives from all you talented people. The list is free of charge to anyone who sends a SASE to me. For a charge of $1.00, I will provide the list on labels. Feel free to make copies and distribute this list to anyone.

If I am not on your mailing list, please add my name and the names of the following VIC 20 owners:

Colin F. Thompson
BASF Systems Corp.
1307 Colorado Ave.
Santa Monica, CA 90404
(213) 451-8781

KEY / SORT /	TEL #	NAME	STREET ADDRESS	CITY / STATE / ZIP
1 KEY LISTING		d: = RETAIL DEALERS		
2 KEY LISTING		h: = HARDWARE MANUFACTURERS		
3 KEY LISTING		p: = PERIODICALS & NEWSLETTERS		
4 KEY LISTING		m: = MAIL ORDER HOUSES		
5 KEY LISTING		s: = SOFTWARE WRITERS		
6 KEY LISTING		dr: = WHOLESALE DISTRIBUTOR		
7 THIS LIST ON LABELS—SEND $1		To: Colin F. Thompson	1307 Colorado Avenue	Santa Monica, CA 98404
d: commo comp ctr		Commodore Computer Center	930 Town & Country Village	San Jose, CA 95128
d: computer barn		Computer Barn	319 Main Street, #2	Salinas, CA 93901
d: computer svc ctr		Computer Service Center	1115 Third Street	San Rafael, CA 94901
d: nat camera	213-829-5465	National Camera & Audio	3223 Wilshire Blvd.	Santa Monica, CA 90049
d: pc comp	415-527-6044	P.C. Computers	10166 San Pablo Ave.	El Cerrito, CA 94530
d: software spot	213-477-7561	The Software Spot	10977 Santa Monica Blvd.	Los Angeles, CA 90025
d: comp specialti		Computer Specialties	1253 Broadway	El Cajon, CA 92021
dr: ics	801-373-2901	ICS Micro Wholesale	Box 1243	Provo, UT 84603
ds: data eq sup	213-923-9361	Data Equipment Supplies	8315 Firestone Blvd.	Downey, CA 90241
h: alphacom		Alphacom	2323 South Bascom	Campbell, CA 95008
h: anvil	213-575-8614	Anvil Cases	4128 Temple City Blvd.	Rosemead, CA 91770
h: buscosys	413-567-8584	Business Comp. Sys. of New England	Box 2285	Springfield, MA 01101
h; carry comp		Carry Comp	24687 Aric Way	Elkhart, IN 46517
h: central	503-244-5782	Central Point Software	Box 19730, #203	Portland, OR 97207
h: comp case	614-868-9464	Computer Case Company	5650 Indian Mound Court	Columbus, OH 43213
h: digital int	503-295-5890	Digital Interface Systems	Box 8715	Portland, OR 97207
h: ecx	415-944-9277	EXC Computer Co.	2678 North Main Street	Walnut Creek, CA 94526
h: glouster		Glouster Computer Bus Co.	6 Brooks Road	Glouster, MA 01930
h: int con	714-641-0181	Integrated Controls	1240-L, Logan Avenue	Costa Mesa, CA 92626
h: parsec res		Parsec Research	Drawer 1766-P	Fremont, CA 94538
h: precision	801-487-6266	Precision Technology	2970 South Richard Street	Salt Lake City, UT 84115
h: ram/rbc		RAM/RBC Systems	Box 351	Malden, MA 02148
h: rvr sys		RVR Systems	Box 265	Dewitt, NY 13214
h: slagh systems	(send SASE)	Slagh System Services	Box 53	Dearborn, MI 48121
h: street elect		Street Electronics Corp.	1140 Mark Avenue	Carpinteria, CA 93013
h: sunshine		Sunshine Peripherals	1229 East 28th Street	Brooklyn, NY 11210
h: torrey	213-247-6486	Torrey Engberg Smith Co.	Box 1075	Glendale, CA 91209
hm: micro-sys	214-484-7836	Micro-Systems Development	11105 Shady Trail, Suite 103	Dallas, TX 75229
hs: cardco	316-685-9536	Cardco	3135 Bayberry	Wichita, KS 67226
hs: comp mkt	609-795-9480	Computer Marketing Services	300 W. Marlton Pike, Suite 26	Cherry Hill, NJ 08002
hs: comp soft	617-961-5700	Computer Software Associates	50 Teed Drive	Randloph, MA 02368
hs: comp works	602-249-0611	Computer Works	2028 West Camelback	Phoenix, AZ 85015
hs: data 20	714-770-2366	Data 20 Corp.	20311 Moulton Parkway, Suite B10	Luguna Hills, CA 92652
hs: elcomp	714-623-8314	Elcomp Publishing	53 Red Rock Lane	Pomona, CA 91766
hs: meta	503-232-1712	Metaresearch, Inc.	1100 SE Woodward	Portland, OR 97202
hs: micro world	303-936-4487	Micro World Electronix	6340 W. Mississippi Avenue	Lakewood, CO 80226

KEY / SORT /	TEL #	NAME	STREET ADDRESS	CITY / STATE / ZIP
hs: oem inc	305-465-9363	O.E.M. Inc.	2729 So. US #1, Suite 12	Fort Pierce, FL 33450
m: aard (cat. $1)	313-669-3110	Aardvark-80	2352 South Commerce	Walled Lake, MI 48088
m: ab	215-822-7727	AB Computers	252 Bethlehem Pike	Colmar, PA 18915
m: alleg		Allegiance Enterprises	868-96th Avenue NE	Blaine, MN 55434
m: avs	800-638-1688	AVS	7566 Main Street	Sykesville, MD 21784
m: cmart	404-981-5939	C Mart	Box 77286	Atlanta, GA 30357
m: comp exp	313-528-1544	Computer Express	Box 569	Troy, MI 48099
m: comp out	800-634-6766	Computer Outlet	1095 East Twain	Las Vegas, NV 89109
m: comp spec		Computer Specialties	1253 Broadway	El Cajon, CA 92021
m: compu sense	316-684-4668	Compu Sense	812 South Lightner	Wichata, KS 67218
m: computer mail		Computer Mail Order	EAST = (800) 233-8950	WEST = (800) 648-3311
m: comstar	805-968-4668	Comstar	Box 1730	Goleta, CA 93116
m: discount	414-231-1696	Discount Software House	Box 93	Winnebago, WI 54985
m: eav software		EAV Software	17 Marble Avenue	Pleasantville, NY 10570
m: ektype	415-489-1532	Ektype Office Systems	1655 Whipple Road	Hayward, CA 94544
m: embassy		Embassy Computer Products	Box 88	Little Neck, NY 11363
m: jmc		JMC	1025 Industrial Drive	Bensonville, IL 60106-1297
m: little wizard	414-273-5468	Little Wizard Distributing	1211 Lambeth Road, Suite 4	Waukesha, WI 53186
m: lyco	800-233-8760	Lyco Computer	Box 10	Cogan Station, PA 17728
m: merlin	213-316-0945	Merlin Enterprises	Box 2876	Torrance, CA 90509
m: micro-ware	201-838-9027	Micro-Ware Distributors	Box 113	Pompton Plains, NJ 07444
m: micrograms	815-965-2464	Micrograms	Box 2146	Loves Park, IL 61130
m: mooseware		Mooseware, Inc.	Box 17868	Irvine, CA 92713
m: mtg	800-343-0854	MTG Technical Sales	281 Needham Street	Newton, MA 02164
m: neeco	617-449-1760	Neeco	679 Highland Avenue	Needham, MA 02194
m: olympic	800-421-8045	Olympic Sales	216 South Oxford Avenue	Los Angeles, CA 95667
m: optomam	916-621-1090	Optomam Consumer Products	Box 1038	Placerville, CA 95667
m: price	800-343-1078	P.R.I.C.E.	67 Teed Drive	Randolph, MA 02368
m: prickly	602-886-1505	Prickly-Pear Software	9822 East Stella Road	Tucson, AZ 85730
m: protecto	312-382-5244	Protecto Enterprizes	Box 550	Barrington, IL 60010
m: queue	800-232-2224	Queue Cat. #11	5 Chapel Hill Drive	Fairfield, CT 06403
m: rde		RDE Services, Games Dept.	3580 Warringham	Waterford, MI 48095
m: sjb		SJB Distributors	10520 Plano Road, Suite 206	Dallas, TX 75238
m: sunrise		Sunrise Electronics	7057 Lompoc Court	Citrus Heights, CA 95610
m: tis		Total Information Services	Box 921	Los Alamos, NM 87544
m: world elec		World Electronics	177-27th Street	Brooklyn, NY 11232
p: byte house	VIC-NIC-NEWS	The Byte House	Box 981	Salem, NH 03079
p: commander	800-426-1830	Commander	PO Box 98827	Tacoma, WA 98498
p: commodore	215-687-9750	Commodore-Microcomp. Magazine	487 Devon Park Drive	Wayne, PA 19087
p: creative comp		Creative Computing Catalog	39 East Hanover Avenue, Dept. HA1X	Morris Plains, NJ 07950
p: foxfire	713-473-6723	Foxfire Systems, Inc.	3811 Newton	Pasadena, TX 77503
p: midnite	217-864-5320	Midnite Software Gazette	635 Maple	Mt. Zion, IL 625499
p: prog inst	919-489-2198	Programmer's Institute	Box 3191	Chapel Hill, NC 27514
p: strictly comm		Strictly Commodore	47 Coachwood Place NW	Calgary, Ata, Canada T3H 1E1
p: vickie-VIC20, 64, Max		VICKIE, John Rosengarten	3822 North Bell Avenue	Chicago, IL 60618
ps: toronto puc		Toronto PET Users Club	381 Laurence Avenue West	Toronto, Ontario, CAN M5M 1B9
s: abacus	616-241-5510	Abacus Software	Box 7211	Grand Rapids, MI 94510
s: academy		Academy Software	Box 9403	San Rafael, CA 94912
s: amer peri		American Peripherals	122 Bangor Street	Lindenhurst, NY 11757
s: automated		Automated Simulations	Box 4247	Mountain View, CA 94040
s: briley		Briley		
s: broderbund	415-456-6424	Broderbund Software	1938 Fourth Street	San Rafael, CA 94901
s: code works	805-683-1585	The Code Works	Box 550	Goleta, CA 93116
s: comm data	313-685-0113	Comm Data Computer House	PO Box 325	Milford, MI 48042
s: computermat	602-855-3357	Computermat	Box 1664	Lake Havasu City, AZ 86403
s: creative	415-9048-9595	Creative Software	201 San Antonio Circle #270	Mountain View, CA 94040
s: dtc		DTC Software	Box 916	Jansville, WI 53547
s: earthware1		Earthware	Box 30039	Eugene, OR 97403
s: french silk		French & Silk Smoothware	Box 207	Cannon Falls, MN 55009
s: harli soft		Harli Software	1740 Garden Briar Court	Thundar Bay RR#2, Ontario, Canada
s: hes		Human Engineered Software	71 Park Lane	Brisbane, CA 94005
s: hypertech		Hypertech	1820 NE 143rd Street, Penthouse 7	Miami, FL 33181
s: interesting	213-328-9422	Interesting Software	21101 S. Harvard Blvd.	Torrance, CA 90501
s: k8		K8 Software	Box 248C	Canton, CT 06019
s: magic		Magic Carpet	Box 35115	Phoenix, AZ 85069

KEY / SORT /	TEL #	NAME	STREET ADDRESS	CITY / STATE / ZIP
s: micro-ed	612-926-2292	Micro-Ed, Inc.	Box 24156	Minneapolis, MN 55424
s: microsignal		Microsignal	900 Embarcadero Del Mar, Unit A	Goleta, CA 93117
s: microspec	214-867-1333	MicroSpec Ltd.	2905 Ports O'Call Court	Plano, TX 75075
s: midwest micr		Midwest Micro Associates	Box 6148	Kansas City, MO 64110
s: mis	408-338-9546	MIS	250 Fern Rock Way	Boulder Creek, CA 95006
s: mw software		MW Software	Box 126	Urbana, IL 61801
s: nufekop	503-878-2113	Nefekop	Box 156	Shady Cove, OR 97539
s: on line		On Line Software	Box 169	S. San Francisco, CA 94080
s: practical		Practical Applications of Cal.	PO Box 255768	Sacramento, CA 95825
s: public	513-698-5638	Public Domain	5025 South Rangeline Road	West Milton, OH 45383
s: qbf	800-547-5995, ext. 194	Quick Brown Fox	548 Broadway, Suite 4F	New York, NY 10012
s: qumax	716-338-2145	Qumax/GRW Laboratories	Box 17010	Rochester, NY 14617
s: rak elect		RAK Electronics	Box 1585	Orange Park, FL 32073
s: random	904-837-7201	Random Access Computers	Box 1453	Benning, FL 32541
s: rapid	413-549-3744	Rapidwriter	91 Long Hill Road	Leverett, MA 01054
s: rar-tech		RAR-TECH	Box 761	Rochester, MI 48063
s: raymac	408-338-9448	RAYMAC Software Group	495 Band Road	Boulder Creek, CA 95006
s: scientivic		Scientivic Software	525 Lohnes Drive	Fairborn, OH 45324
s: skyles	415-965-1735	Skyles Electric Works	231E South Whisman Road	Mountain View, CA 94041
s: specific	408-241-0181	Specific Software	Box 10516	San Jose, CA 95157
s: taylor	402-464-9051	Taylormade Software	8053 East Avon Avenue	Lincoln, NE 68505
s: telegames		Telegames	RR#1, Hampton, Box 152	Ontario, Canada LOB 1J0
s: thorn-emi		Thorn-EMI		
s: totl	415-943-7877	TOTL Software	Box 4742	Walnut Creek, CA 94596
s: transonic	507-387-1642	Transonic Laboratories	249 Norton St.	Mankato, MN 56001
s: tsasa	609-346-3063	TSASA	2 Chipley Run	West Berlin, NJ 08091
s: tyrant		Tyrant Software	Box 31569	Aurora, CO 80041
s: victory	215-576-5625	Victory Software	2027-A SJ Russell Circle	Elkins Park, PA 19117
s: west ne		Western New England Software	Box 31	Wilbraham, MA 01095
s: wil. robbins		William Robbins	Box 3745	San Rafael, CA 94912
s:wunder	503-899-7549	Wunderware	Box 1287	Jacksonville, OR 97530
sp: avalon	301-254-5300	Avalon Hill Games	4517 Hartford Road	Baltimore, MD 21214

New Products

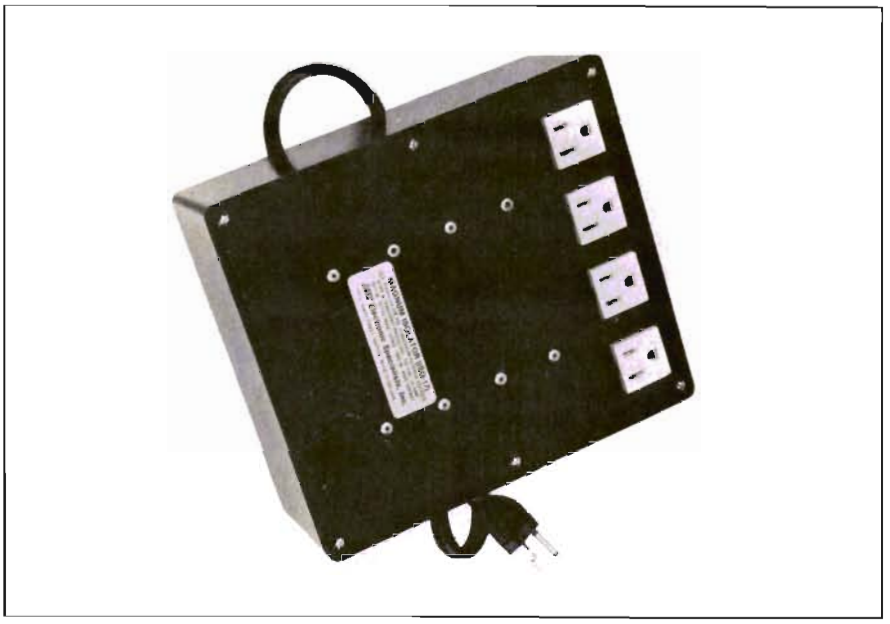
VANILLA PILOT

TAMARACK SOFTWARE has announced the release of **Vanilla Pilot**—a plain vanilla version of the PILOT computer language at an unbelievably low price. The language includes a set of powerful additions to the resident editor in the computer. It is a full-featured Pilot language including TURTLE GRAPHICS. **Vanilla Pilot** will be available for the Commodore 4000, 8000, 9000, and the Commodore-64.

The **Vanilla Pilot** editor is used in conjunction with the screen editor of the computer. The editor adds a number of features which will permit easy program entry and debugging. It has 19 commands including FIND/CHANGE, TRACE and convenient disk and cassette input/output commands. The TRACE command will list the Pilot statement currently being executed in a line at the top of the screen. The disk LOAD command features an append option. The disk SAVE command includes a partial save. RUN includes an option to load a program from disk and begin execution.

A special feature of the interpreter section is a full TURTLE GRAPHICS package. With this, the user can control the turtle's DIRECTION and place the pen UP or DOWN. In the Commodore-64 version the pen can be set to any one of the 16 screen colors available. The turtle can DRAW lines and turn to the LEFT or RIGHT.

Another section of the interpreter has a multi-featured screen command. With this screen command you can perform any of the cursor movements or switch between uppercase/graphics and upper/lower case screen displays. In addition, you can set line spacing to single or double space on the screen displays. Or, you can reverse the entire screen for dark characters on a light background.



The manual which accompanies **Vanilla Pilot** was written by experienced educators and was carefully designed for clarity and easy reading. It is fully illustrated and contains a number of programming examples. There also is an appendix with all of the statements and commands clearly documented so that an experienced programmer can immediately begin using PILOT.

Vanilla Pilot will be available from your local Commodore computer dealer in December, 1982 with a suggested retail price $29.95.

Tamarack Software, Darby, MT 59829, (406) 821-3924.

SOFTWARE POLLUTION CONTROL

Electrical pollution drives Micro Programs bananas! Power line electrical noise, hash and spikes often cause erratic computer operation. In addition, severe spikes from lightning or heavy machinery may damage expensive hardware.

Many systems create their own pollution! Disks and printers often create enough electrical interference to disrupt an entire program. Nearby electronic equipment is affected as well.

Electronic Specialists recently announced **Magnum Insolator** is designed to control severe electrical pollution. Incorporating heavy duty spike/surge suppression, the **Magnum Isolator** features four individually quad-Pi filtered AC sockets. Equipment interactions are eliminated and disruptive/damaging power line pollution is controlled. The **Magnum Isolator** will control pollution for an 1875 watt load. Each socket can handle a 1000 watt load.

The **Model ISO-17 Magnum Isolator** eliminates severe AC power line pollution for smooth program operation. $181.95.

ELECTRONIC SPECIALISTS, INC., 171 South Main Street, P.O. Box 389, Natick, Mass. 01760. Phone (617) 655-1532.

Hayes Microcomputer Products, Inc.

Norcross, Georgia-Hayes Microcomputer Products, Inc. announces Smartcom II™, communications software for the IBM Personal Computer with a Hayes Smartmodem 300 or high speed Smartmodem 1200. Smartcom II manages data transfer over the telephone lines and brings the microcomputer, disk drives, and printer into the activity.

Smartcom II extends remote computing to a wide range of microcomputer users. To aid the new user, Smartcom II is built around a simple but comprehensive menu of program options, supported by "Help" information displayed on demand. The Help feature provides a quick response to questions about parameters, prompts, and messages. To satisfy the more demanding user, Smartcom II transfers program files error-free and allows the Smartmodem to be tailored for a unique communications environment.

President of Hayes Microcomputer Products, Inc., Dennis C. Hayes commented, "Smartcom II, our second software release, continues our commitment to developing high-quality, easy-to-use programs for our products. Smartcom II takes full advantage of the Smartmodem's conveniences and offers further labor-saving, time-saving, and thus cost-saving features for on-line use."

Supported by the auto-dial/auto-answer Smartmodems, Smartcom II automatically originates and answers telephone calls. It automatically logs a user onto a remote system, such as a time-sharing service, information utility, data base, or microcomputer. To save the user the time and trouble of rekeying a sequence of commands or information regularly sent to another computer, Smartcom helps a user to compose and store this information as a Macro. One Macro is reserved for the automatic log-on, the others are executed on the remote computer by just two keystrokes. Both save connect time and money. The Smartcom disk comes prepared with Macros for The Source, CompuServe, and Dow Jones information services.

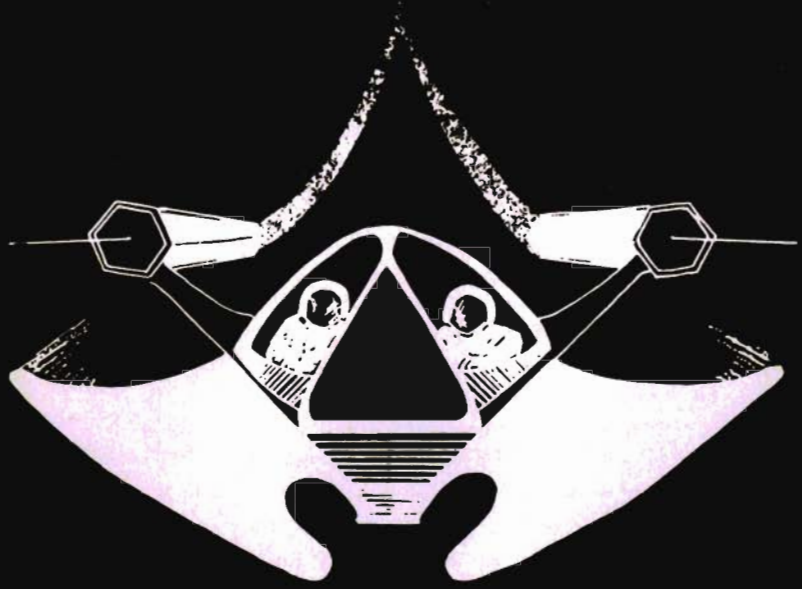
To eliminate repetitive changes to a single set of program options, Smartcom II lets a user select and store parameters and up to twenty-six Macros for each remote system called. Each group of parameters and Macros makes up a Communication Set, added by the program to a personal Communication Directory. Communications parameters include telephone number, baud rate, duplex, character delay, confidential mode, password, keyboard definitions, and others.

Smartcom II captures incoming data to disk and printer concurrent with its display on the screen. Special keys that stop and start the data capture allow selective storing and printing of data. File transfer is managed by three different protocols: Stop/Start, Send Lines, and the Verification Protocol for error-free transmission between Hayes programs (including the Hayes Terminal Program for the Micromodem II™ and Apple II). A Remote Access feature provides the originator of a call with the ability to send and receive files form an unattended system running Smartcom II.

For further ease, Smartcom II displays a Disk Directory and creates, displays, prints, erases, and renames files without returning to the operating system, even while connected with another computer. One keystroke toggles between the remote screen and local menu.

The program supports up to sixteen disk drives (including a hard disk), both parallel and serial printers, and either the monochrome or color/graphics display. The program requires an eighty column monitor, one disk drive, 96K RAM, an asynchronous communications card, and DOS 1.10 or 1.00. Estimated retail price for Smartcom II, with its complete owner's manual, is $119.00. Smartcom II for the IBM Personal Computer will be available through retail computer stores in February.

GAME PROGRAM DEVELOPMENT KIT



for the

COMMODORE VIC - 20

SIX TOOLS TO HELP YOU WRITE YOUR OWN FAST ACTION ARCADE-STYLE GAMES

DECODER — Decodes programs written in machine language (like game cartridges, utility cartridges, and even the computer's own internal operating programs). Produces a program in an English-like language (Assembler) which can be studied to figure out how they did it. The programs created with the decoder can be customized with the **EDITOR** AND INCORPORATED INTO YOUR OWN NEW GAME PROGRAM. The **ASSEMBLER** turns your programs created with the Decoder and the Editor back into machine language and puts them out to tape or disk so the **LOADER** can load them into the computer's memory to be tested and RUN. The **MONITOR** assists you in debugging your new game program by allowing you to run it a step at a time and making modifications if you need to. The **INSTRUCTION GUIDE** is written so that even a beginner can learn the skills needed to become a pro!!!

DESIGNED TO RUN ON ALL VIC-20's

$49.95 plus $2.00 p&h buys the kit that could make you rich. Why wait?

Send check, M.O., VISA/MC ($2.00 s.c., please include expiration date), or specify COD (add $3.00) to:

French Silk smooth ware

P.O. Box 207-C, Cannon Falls, MN 55009

507-263-4821

Single Drive Copy

by Howard Rotenberg
Ontario, Canada

I was sitting at my computer thinking about the possible problems on copying files on a single disk drive. Luckily I do not have to do this but there are more users now with a single disk drive. I thought it would be an interesting project, so I started to think about tackling it. After a little more thinking I realized that keeping track of a pointer into the file was the key to the program. Since BASIC doesn't support pointers as such, I decided an index to a temporary buffer which I would use to hold the partial file during transposition was the answer. This would act as my pointer and now I could start to write the code.

The principal of the program is actually quite easy. The idea is to read the source file until either the buffer is full or you reach the end of file. When one of these conditions are met you must then put in the destination diskette. The catch here is that the complete file may not fit into the buffer. Here is where the pointer comes into play.

As we index into the buffer, we increment both a buffer index and a ongoing pointer. This pointer will be used at a later time. As mentioned before, the destination diskette is put in the drive and the file written until the buffer is completely dumped. The append command came in very handy for this program. The source diskette is then put back into the drive. (FOR THE CURIOUS): This is where the pointer finally comes in. We now have to read the source diskette again; but how much of it? Since I kept an ongoing pointer, a subroutine that just reads bytes until the value of the pointer is reached is used. The reads as such: FOR BYTE = 1 to PTR%: GET#1,BYTE$: NEXT: RETURN. This just reads into the file up to the point that we left off last time. Now we can go back to the regular routine and continue to fill the buffer with more of the file.

This whole process continues until the complete file is copied. The number of times you must switch disks depends on the size of the buffe use and the actual file size, of co In trying to use the largest buffe possible, I found out that every t byte is saved, three bytes of me are used. This is a limiting factor f buffer size.

I used variables and string n that I hope are fairly self docume This program only copies sequ files properly at the moment. T would have to be some changes a small conversion program to for a regular program. A small n remember is that when the outp is initially opened there is a chr left as the first byte. Although routine was written for the Commo computer (BASIC 4.0), I believe the program may be easily cha for any computer. In conclus would like to say that after testing using this program, with all its dis changes; "THANK GOODNESS DUAL DISK DRIVES."

```
READY.
 10 REM  ************************************
 20 REM  *        SINGLE DRIVE COPY         *
 30 REM  *          FOR BASIC 4.0           *
 40 REM  * THIS PROGRAM COPYS SEQUENTIAL    *
 50 REM  *  FILES FROM A SOURCE DRIVE TO     *
 60 REM  *  THE SAME DESTINATION DRIVE.      *
 70 REM  *  GOOD FOR SINGLE DISK DRIVES      *
 80 REM  *     BUT MAY BE USED ON DUAL       *
 90 REM  *                                   *
 100 REM *       BY HOWARD ROTENBERG         *
 110 REM *         TORONTO ONTARIO           *
 120 REM ************************************
 130 BUFFER = 4096
 140 DIM SVBYTE$(BUFFER)
 150 INDEX% = 0
 160 PTR% = 0
 170 OPEN15,8,15
```

```
180 BB$ = " DISKETTE AND PRESS ← WHEN READY"
190 D2$ = "WRITING TO DESTINATION"
200 F1$ = "PUT IN SRC"
210 F2$ = "PUT IN DEST"
220 D1$ = "READING SRC"
230 INPUT"INPUT FILE";F$
240 INPUT"OUTPUT FILE";G$:
250 PRINT"THE BUFFER CAN HOLD"BUFFER/256"BLOCKS"
260 PRINTF2$BB$" TO OPEN NEW FILE"
270 GOSUB1070
280 REM
290 REM ************************************
300 REM *  OPEN INITIAL FILE TO COPY TO *
310 REM ************************************
320 REM
330 DOPEN#1,(G$),W
340 GOSUB 1130
350 PRINTF1$BB$
360 CLOSE1
370 GOSUB1070
380 REM
390 REM ************************************
400 REM *   READ FILE AND FILL BUFFER   *
410 REM ************************************
420 REM
430 DOPEN#1,(F$)
440 GOSUB 1130
450 IF PTR% THEN GOSUB 950
460 GET#1,BYTE$
470 SVBYTE$(INDEX%) = BYTE$
480 INDEX% = INDEX% + 1
490 PTR% = PTR% + 1
500 IF ST <> 64 AND INDEX% <= BUFFER - 1 THEN 460
510 IF ST = 64 THEN CLOSE1:GOTO790
520 CLOSE1
530 REM
540 REM ************************************
550 REM *    WRITE OUT BUFFER TO FILE   *
560 REM ************************************
570 REM
580 PRINTF2$BB$
590 GOSUB1070
600 DD$ = D2$
610 GOSUB1100
620 APPEND#2,(G$)
630 GOSUB 1130
640 FOR BYTE = 0 TO INDEX%
650 PRINT#2,SVBYTE$(BYTE);
660 NEXT
670 CLOSE2
```

```
680 INDEX% = 0
690 PRINTF1$BB$
700 GOSUB1070
710 DD$ = D1$
720 GOSUB1100
730 GOTO 430
740 REM
750 REM************************************
760 REM*   DUMP LAST REMAINS OF BUFFER    *
770 REM************************************
780 REM
790 PRINTF2$BB$" FOR LAST DUMP"
800 GOSUB1070
810 DD$ = D2$
820 GOSUB1100
830 APPEND#2,(G$)
840 GOSUB 1130
850 FOR BYTE = 0 TO INDEX% - 1
860 PRINT#2,SVBYTE$(BYTE);
870 NEXT
880 DCLOSE: PRINT"COPY FINISHED": END
890 REM
900 REM ************************************
910 REM * READ FILE UP TO LAST POSITION *
920 REM *     AND THROW AWAY ALL BYTES     *
930 REM ************************************
940 REM
950 FOR BYTE = 1 TO PTR%
960 GET#1,BYTE$
970 NEXT
980 RETURN
990 REM
1000 REM ************************************
1010 REM *         THREE ROUTINES TO        *
1020 REM *     FLASH CURSOR DURING GET      *
1030 REM *           INITIATE DISKS         *
1040 REM *     AND CHECK FOR DISK ERROR     *
1050 REM ************************************
1060 REM
1070 POKE 167,0: GET C$:IF C$ <> "←" THEN 1070
1080 POKE 167,1
1090 RETURN
1100 PRINTDD$
1110 PRINT#15,"I0"
1120 RETURN
1130 IF DS THEN PRINT DS$: DCLOSE: END
1140 RETURN
READY.
```

RADIX-50: Pack & Unpack

by Howard Rotenberg
Ontario, Canada

Have you ever been stuck for memory? Have your disk files just seemed to grow like weeds? If you answered yes to either one of these questions then RADIX-50 may be for you. By now you may be wondering what this RADIX-50 that sounds like something out of a sci-fi movie is all about. I can assure you that it is not science fiction nor is it a new element of nature, although it has been around for quite some time.

To put it simply, RADIX-50 is a way to convert or pack 24 bits of information into 16 bits or one word. This means a saving of ⅓ memory from the original input. Before I go into the details of this handy little application, let's take a brief look at the history behind RADIX-50.

RADIX-50 was developed for the PDP-8 computer approximately 15-20 years ago. (FOR THE TRIVIA NUTS), the PDP-8 was a successor of the PDP-1 that was developed in Chalk River, Canada. The reason for the implementation of RADIX-50 will become very obvious shortly. The PDP-8 had a symbolic assembler with 8, yes only 8 instructions. The symbols of course took up memory and (HERE'S THE CLINCHER) there was only 4k of memory available at the time. This, as you could imagine, led to problems when it came to memory management. The answer was RADIX-50.

The chance to decrease the amount of memory needed for program development was too good to pass up. There was one major restriction to this method; only 40 characters were allowed. The valid characters were the alphanumerics, a space, two additional characters such as the dollar sign or an ampersand and one special character that the user could choose. The later algorithms allowed the pro-

```
READY.

 10 REM*********************************
 20 REM*            RADIX-50           *
 30 REM*  IMPLEMENTED ON A COMMODORE    *
 40 REM*  BUT MAY BE EASILY MODIFIED    *
 50 REM*  FOR ANY BASIC OR HIGH LEVEL   *
 60 REM*    LANGUAGE ON ANY COMPUTER    *
 70 REM*                                *
 80 REM*       BY HOWARD ROTENBERG      *
 90 REM*         TORONTO ONTARIO        *
 100 REM*********************************
 110 REM
 120 REM*********************************
 130 REM*  THIS ROUTINE SETS THE 3       *
 140 REM* NON-ALPHANUMERIC CHARACTERS     *
 150 REM*  THAT WILL BE ACCEPTED AS      *
 160 REM* INPUT BY THE PACKING AND       *
 170 REM*   UNPACKING SUBROUTINES        *
 180 REM*********************************
 190 REM
 200 PRINT "□ENTER THE 3 SPECIAL CHARACTERS
     TO BE"
 205 PRINT"ACCEPTED"
 210 INPUT N$
 220 FOR J = 1 TO 3
 230 SP$(J) = MID$(N$,J,1)
 240 NEXT
 250 REM
 260 REM *********************************
 270 REM * ACCEPT THREE ASCII BYTES      *
 280 REM *********************************
 290 REM
 300 PRINT: PRINT"□NOW ENTER THE CHARACTERS
     TO BE PACKED"
 305 PRINT"INDIVIDUALLY"
 310 PRINT"REMEMBER ONLY 0-9,A TO Z,SPACE
     AND "SP$(1)" "SP$(2)" "SP$(3);
 320 PRINT"ARE VALID CHARACTERS"
 330 PRINT"TO ENTER A SPACE JUST
     PRESS RETURN"
 340 GOSUB 710
 350 C1 = N
 360 GOSUB 710
```

grammers three special characters of their own choice. Why was this feature called RADIX-50 when all you have been hearing about is the restricted use of 40 characters? The answer is simple for those of you who have not already guessed. The PDP-8 used the octal number system and 50 octal converts to 50 decimal.

Now we can get into the inner workings of the algorithm. I have translated it for the Commodore computer. The method used is actually quite simple. The first work that has to be done is to take the ascii character and convert it to tis RADIX-50 code representation. This is done by subtracting the appropriate amount from the input character. The final numeric code will be from 0 to 39 inclusive. For the characters a to z (decimal 65 to 90) we will subtract 64. We subtract 21 from the numerics (decimal 48 to 57). This takes care of the codes 1 through 36. Lastly we set the space to 0, and the three special characters to 37, 38 and 39 consecutively. It is important to remember that we must always work with a set of triplets (three characters) for the algorithm to work efficiently. As the codes were set they were assigned to variables to prepare for a quick calculation to pack them into one word. In the sample program I have included, the variables are: C1, C2, and C3. The equation for the packing is: C4 = ([C1 * 40] + C2) * 40 + C3, with C4 now holding the word that contains our information. For example, if we save the characters A, B and C our formula would translate to C4 = ([1 * 40] + 2) * 40 + 3. The variable C4 would be equal to 1683. We have just put three bytes into one word or two bytes. This may not seem like such a big deal at this time but it now allowed a 6000 byte program to reside in a computer with only a 4000 byte capacity. It has cut the memory needed for any given program by ⅓. Of course the savings increase with the order of magnitude of your programs. Consider your 32k file or 32k array now only needing approximately 21.3k bytes of memory. It is now that you may start thinking how useful this feature may be.

Now that we have all this informa-

```
370 C2 = N
380 GOSUB 710
390 C3 = N
400 REM
410 REM ********************************
420 REM * PACK THE THREE CHR'S INTO  *
430 REM *          ONE WORD            *
440 REM ********************************
450 REM
460 C4 = ((C1*40)+C2)*40+C3
470 PRINT"THE THREE CHARACTERS ARE NOW
    PACKED INTO THE WORD"C4
480 REM
490 REM ********************************
500 REM * UNPACK THE RADIX-50 WORDS  *
510 REM ********************************
520 REM
530 PRINT"■THE UNPACKED CHARACTERS ARE ";
540 N = INT(C4 / 1600)
550 GOSUB 1000
560 N = INT((C4 - (C1 * 1600)) / 40)
570 GOSUB 1000
580 N = C4 - (C1 * 1600) - (C2 * 40)
590 GOSUB 1000
600 PRINT
610 INPUT"■CONTINUE OR END   C■■■";C$
620 IFC$="C"GOTO300
630 END
640 REM
650 REM ********************************
660 REM *  CONVERTS AN ASCII CHR TO *
670 REM *      ITS RADIX-50 CODE      *
680 REM *        REPRESENTATION       *
690 REM ********************************
700 REM
710 INPUT"    ■■■";N$
720 N = ASC(N$)
730 IF N = 160 THEN N = 0:RETURN
740 IF N >= 65 AND N <=90 THEN N = N - 64:
    RETURN
750 IF N >= 48 AND N <=57 THEN N = N - 21:
    RETURN
760 IF N$ = SP$(1) THEN N = 37:RETURN
770 IF N$ = SP$(2) THEN N = 38:RETURN
780 IF N$ = SP$(3) THEN N = 39:RETURN
790 PRINT"■NOT ONE OF INDICATED
    SPECIAL CHR'S"
800 PRINT"■■I HAVE SET IT TO A SPACE■"
810 N = 0:RETURN
820 REM ********************************
830 REM *  CONVERT FROM RADIX 50    *
840 REM * AND RETURN ITS EQUIVILENT *
850 REM *         ASCII CHR          *
860 REM ********************************
```

tion neatly stored it is time to get it back into its original form. This is actually the inverse operation to the packing of the bytes. To begin with we must unpack the RADIX-50 words individually. This is done with these formulas:

Character #1 = INT(C4 / 1600)
Character #2 = INT([C4 – (C1 * 1600)] / 40)
Character #3 = C4 – (C1 * 1600) – (C2 * 40)

For the computers that have the mod function the decoding is a little easier:

Character#1 = INT(C4 / 1600)
Character #2 = INT(C4 / 40 MOD 40)
Character #3 = C4 MOD 40

These formulas will return us the RADIX-50 representations consisting of the numbers between 0 and 39 inclusive. This, as you may recall, is exactly what we originally converted our characters down to. We now convert from RADIX-50 and return its equivalent ascii character. This is done exactly opposite from the way we converted to the code to begin with. For the numerics 1 to 26 we add 64 which will give us the ascii characters a to z. For the numerics 27 to 36 we will add 21 which will retrieve our ascii digits 0 to 9. If the numeric is 0 we set the character to a space. Lastly if the numerics are 37, 38 or 39 we return the special character that it was set to by the program. Using the above formulas and the original characters we used to pack down, this is what we would come up with. Recall that C1 = A, C2 = B, C3 = C and the packed word was 1683.

X = INT(1683 / 1600)
X is equal to 1 and after adding 64, chr$(n) = a
Y = INT([1683 – (1 * 1600)] / 40)
Y is equal to 2 and after adding 64, chr$(n) = b
Z = 1683 – (1 * 1600) – (2 * 40)
Z is equal to 3 and after adding 64, chr$(n) = c

You can see that we now have our original three characters back.

The sample program I have included allows you to experiment with RADIX-50. I use one safe-guard for the input of your characters. If you enter an invalid character, I set it to a space and inform you that this is the case. I have also allowed you to key in a return for a space although it is really accepting a shifted space. This may clear up the reason for checking for a 160 instead of an ascii 32. In using this for real applications you would of course change this to a 32 (ascii space). On other computers you may have to change some of the ascii checks depending on what asc(a$) will return. This may even be true on the earlier Commodore computers since this program was done on an 8032.

```
 870 REM
 880 REM
 890 IF N = 0 THEN N = 32:RETURN
 900 IF N >= 1 AND N <=26 THEN N = N + 64:
     RETURN
 910 IF N >= 27 AND N <= 36 THEN N = N
     + 21:RETURN
 920 IF N  = 37 THEN  N$ = SP$(1):N = 0
     :RETURN
 930 IF N  = 38 THEN  N$ = SP$(2):N = 0
     :RETURN
 940 IF N  = 39 THEN  N$ = SP$(3):N = 0
     :RETURN
 950 REM
 960 REM ********************************
 970 REM *  GET CHR THEN DISPLAY IT  *
 980 REM ********************************
 990 REM
1000 GOSUB 890:IF N THEN PRINTCHR$(N);
     :RETURN
1010 PRINTN$;:RETURN
READY.
```

SUMMARY:

So what good is all this to me? The applications are only limited to your imagination. You may write sequential files using the packed format. This not only decreases your disk space needed but is a form of protection. If any one tries to read your files, all they see is a lot of meaningless numbers. It goes without saying that when you read your files you will have to translate them back to ascii. This of course is no problem since you already have the means available to you. Another saving you may use is to store your strings as words in memory. This you will find will give you a substantial amount of memory savings. The only real criteria is that you stick to the restricted characters and the Commodore computers don't like a quote or colon for the special characters. If your string does not work out to a number of bytes divisable by three, the algorithm will still work but you will waste a byte since either one or two bytes will still get packed into a word.

I hope this will intrigue some of you into using this old but still useful technique. Hopefully we will see some of your applications in the near future, but until then I think I will just PACK it in.

COMPU SENSE

VIC-20®

VIC-20®	**Personal Computer**	$179.95
VIC-1011A	**RS232C Interface**	39.95
VIC-1515	**Printer**	334.95
VIC-1530	**Datasette**	67.50
VIC-1540	**Disk Drive**	349.95
VIC-1010	**Expansion Module**	139.95
VIC-1311	**Joystick**	9.95
VIC-1312	**Game Paddles**	19.95
VIC-1600	**Telephone Modem**	99.95
CM-151	**Terminal 40**	29.95

Produces 40 column output of information received through the modem

VIC-1210	**VIC 3K Memory Expander Cartridge**	34.95

Plugs directly into the VIC's expansion port. Expands to 8K RAM total.

VIC-1110	**VIC 8K Memory Expander Cartridge**	52.50

8K RAM expansion cartridge plugs directly into the VIC

VIC-1011A	**RS232C Terminal Interface**	39.95

Provides interface between the VIC-20 and RS232 telecommunications modems. Connects to VIC's user port.

CM	**16K Memory Expander**	89.95
CM	**24K Memory Expander**	110.95

CARDBOARD 6 $87.95
An expansion interface for the VIC-20. Allows expansion to 40K or accepts up to six games. May be daisy chained for more versatility.

CARDBOARD 3 $29.95
Economy expansion interface for the VIC-20

CARD "?" CARD/PRINT $79.95
Universal Centronics Parallel Printer Interface for the VIC-20 or CBM-64. Use an Epson MX-80 or OKIDATA or TANDY or just about any other

CARDETTE $39.95
Use any standard cassette player/recorder with your VIC-20 or CBM-64

CARDRITER $29.95
A light pen with six good programs to use with your VIC-20 or CBM-64.

BUSINESS & HOME APPLICATIONS FOR VIC-20®

CW-107A	**Home Calculation Program Pack**	$48.95
CPV-31	**Data Files** - your storage is unlimited	14.95
CPV-96	**Household Finance Package** - to keep records of all your household expenses	30.95
CPV-208	**Bar-Chart** - display your numerical data	8.95
CH	**Turtle Graphics** - learn programming	34.95
CH	**VIC Forth** - is a powerful language for BASIC programming	49.95
CH	**HES MON** - is a 6502 machine language monitor with a mini-assembler	34.95
CH	**HES Writer** - time-saving word processing tool	34.95
CH	**Encoder** - keep your personal records away from prying eyes	34.95
CT-21	**Statistics Sadistics** - statistical analysis	14.95
CT-121	**Total Time Manager 2.0** - creates personal or business schedules	15.95
CT-124	**Toll Label** - a mailing list and label program	13.95
CT-125	**Toll Text BASIC**	15.95
CT-126	**Research Assistant** - keep track of reference data	17.50
CT-140	**Toll Text Enhanced**	29.95
CM-152	**Grafix Designer** - design graphic characters	12.95
CQ-5	**Minimon** - allows you to program, load, save, or execute machine language programs	13.95
CT-3	**Order Tracker**	15.95
CT-4	**Business Inventory** - to maintain record of inventory	15.95
CPV-210	**Bidder**	13.95
CPV-217	**Cash Flow Model** - determine cash flow	12.95
CPV-220	**Client Tickler**	16.95
CPV-221	**Club Lister**	13.95
CPV-224	**Depreciator**	9.95
CPV-236	**Investment Analyst** - keep track of investments and investment opportunities	12.95
CPV-251	**Present Value**	10.95
CPV-269	**Super Broker**	12.95
CPV-270	**Syndicator** - calculates whether to buy or sell	13.95
CPV-274	**Ticker Tape** - maintains investments profile	14.95
CPV-276	**Un-Word Processor** - screen editor	16.95
CPV-286	**Phone Directory** - never lose a phone number again	9.95
CS-111	**Checkbook** - home "utility" program	14.95
CPV-294	**Calendar My Appointments** - print a calendar for every month in any year.	14.95
CPV-296	**The Budgeter** - place your personal finances in order	12.95
CPV-327	**HESCOM** - transfers data and programs bidirectionally between VICs at three times the speed of a disk drive	40.95
CPV-328	**HESCOUNT** - monitors program execution	19.95
CHV	**HESPLOT** - Hi-res graphics subroutines	12.95
CPV-367	**Conversions** - figures, volume, length, weight, area, and velocity to all possible configurations	7.95
CC	**The Mail** - your complete mail program — Cassette	24.95
	Disk	29.95
CS	**Home Inventory** - lists your home belongings	17.95
CS	**Check Minder** - (V-20 & 64) keep your checkbook the right way	14.95
CS	**General Ledger** - a complete general ledger	19.95

BUSINESS & HOME APPLICATIONS FOR C-64

CHC-504	**HES Writer** - word processor	$39.95
CHC-503	**Turtle Graphics II** - utilizes the full graphics of your 64	49.95
CHC-502	**HESMON** - machine language monitor w/mini-assembler	34.95
CHP-102	**6502 Professional Development System**	29.95
CFC	**Data Files** - a management program	27.95

MANY MORE PROGRAMS FOR YOUR 64 & 20

COMMODORE SOFTWARE

VIC-1211A	**VIC-20 Super Expander**	$57.99

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. 1024x1024 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.

VIC-1212	**Programmer's Aid Cartridge**	$45.99

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

VIC-1213	**VICMON Machine Language Monitor**	$48.99

Helps machine code programmers write fast efficient 6502 assembly language programs. Includes one line assembler/disassembler.

GAMES FOR YOUR VIC-20®

CCS	**Cribbage**	$14.95
CCD	**Motor Mouse**	12.99
CW-1901	**Avenger Cart.** - an invasion of space intruders and you're the VIC "Avenger"	24.95
CW-1904	**Supersiot Cart.** - great music and sound effects!	24.95
CW-1906	**Super Alien Cart.** - you're trapped in a maze	24.95
CW-1907	**Jupiter Lander Cart.** - pilot your "Jupiter Lander"	24.95
CW-1908	**Draw Poker Cart.**	24.95
CW-1909	**Midnight Drive Cart.** - authentic night driving	24.95
CW-1910	**Radar Rat Race**	24.95
CW-1911	**Sky Falling**	24.95
CW-1912	**Mole Attack** - a colorful "cartoon action" game	24.95
CW-1913	**Raid on Ft. Knox** - try to escape the guards	24.95
CW-1914	**Adventure Land** - Formerly available only on larger, more expensive computers. All Adventure games are decoded to "talk" on the Type N Talk voice synthesizer (available from VOTRAX)	31.95
CW-1915	**Pirate Cove Adventure** - Yo, ho, ho, & a bottle of rum	31.95
CW-1916	**Mission Impossible Adventure**	31.95
CW-1917	**The Count Adventure** - trapped in Dracula's castle with 3 days to find and destroy the vampire	31.95
CW-1918	**Voodoo Castle Adventure** - you have to free Count Yorga from a curse	31.95
CW-1919	**Sargon II Chess** - seven challenging play levels	31.95
CW-1923	**Gorf** - (The smash-hit arcade game!)	31.95
CW-1924	**Omega Race** - the ultimate space game	31.95
CW-1937	**Seawolf** — an explosive Bally Midway arcade "classic"	24.95
CH-G202	**Maze of Mikor** - adventure-packed game with stunning graphics	15.95
CH-G203	**Tank Wars**	15.95
CH-G205	**Pinball**	13.45
CH-G206	**Simon** - It gets tougher as you get better. Great for kids of all ages.	13.45
CH-G207	**Fuel Pirates**	13.45
CH-G209	**Laser Blitz**	15.95
CH-G210	**Tank Trap**	15.95
CH-G211	**Concentration**	13.45
CH-G212	**Dam Bomber** - pilot your plane, avoid enemy fire	13.45
CH-C307	**Shamus** - search room after room for the shadow-eluding androids, two levels of intense arcade action	34.95
CH-C308	**Protector**	36.95
CPU-79	**Breakout**	7.95
CPU-85	**Hangman** - unbelievable graphics and sound	9.95
CPU-87	**Memory** - VIC challenges your memory	9.95
CPU-88	**Match** - hand and eye coordination	7.95
CPU-89	**Monks** - a devilish game of logic	7.95
CPU-108	**Bomber** - you must decide who you want to fly for, then pick a target and your experience level	9.95
CPU-109	**Amok** - the halls of Amok are populated by robots that obey one instruction - get the intruder	20.95
CPU-153	**Tank vs. UFO** - the tank is moving back and forth along the base; shoot the UFO before it shoots you	9.95
CPU-194	**Snakman** - Pacman for the VIC	14.95
	Defender on Tri - you're the pilot of the experimental ship, Defender	17.95
	3-D Man - the popular arcade game, requires 3K	17.95
	Exterminator - a game full of bugs	20.95

GAMES FOR YOUR 64

CCS	**Cribbage**		$17.95
CFC	**Flight 64** - what a program!	Cassette	14.95
		Disk	16.95
CFC	**Spright Generator**	Cassette	15.95
		Disk	17.95
	Mastermind (VIC-20) $12.95	(C-64)	19.95
	Star Trek (VIC)		9.95
	Black Jack (VIC)		11.95
	Tic-Tac-Toe		7.95
	Backgammon		14.95
	Maze 64		15.95

Prices subject to change.

TO ORDER:
P.O. Box 18765
Wichita, KS 67218
(316) 684-4660

MasterCard VISA

Personal checks accepted (Allow 3 weeks)
or C.O.D. (Add $2) *Handling charges $2.00*

VIC-20® is a registered trademark of Commodore

COMAL: A Closer Look

by Howard Rotenberg
Ontario, Canada

The first and last time I saw an article about the language COMAL was in the December issue of Compute way back in 1981. It was an excellent overall view presented to us by Jim Butterfield. I have always been in support of structured languages such as Pascal, Waterloo Structured Basic and a few others not available for Commodore computers yet. Since COMAL falls into this category, I chose to discuss the language in a little more depth mainly for the newcomers to structured programming.

To briefly recap a little history about COMAL, it is a public domain program. Yes, this means it is FREE; and currently there are 3 different versions available. The language orignated in Denmark and was expended by Mogens Kjaer for the CMB computer. COMAL first came into existence in 1974. In my opinion it seems to be a cross between Basic and Pascal with a strong flavor of Waterloo Structured Basic. To get a copy just as ask around any PET user groups or possibly your neighborhood computer store will let you copy a version. If you are really interested in all the facts on COMAL you may purchase a kit that contains disks, documentation, binder and a newsletter subscription. This should be available at most Commodore dealers at this time. If all else fails you can write the COMAL Users Group, 5501 Groveland Terrace, Madison, WI 53716.

The three versions of COMAL that I spoke about are COMAL-80, COMAL-80+ and a split version that uses an editor and run time interpreter. Incidently, the split version will fit into a 16k computer and allows a minimum of source codes.

When first entering code using COMAL you may enter a statement such as: FOR J = 1 to 10. Remarkably

```
0010 //*****************************
0020 //*         FLIP-FLOP         *
0030 //* SAMPLE COMAL-80 PROGRAM *
0040 //*    BY HOWARD ROTENBERG    *
0050 //*     TORONTO ONTARIO       *
0060 //*****************************
0070 //
0080 // INITIALIZE DIMENSION STATEMENTS //
0090 //
0100 DIM FLAG(10)
0110 DIM DOWN16$ OF 16, SPACE$ OF 40, LINE$
     OF 37, SECONDLINE$ OF 37
0120 DIM TEMP$ OF 37
0130 DIM CLEARSCREEN$ OF 1, HOME$ OF 1,
     UP$ OF 1, DOWN$ OF 1
0140 //
0150 // INITIALIZE STRINGS //
0160 //
0170 DOWN16$:="■■■■■■■■■■■■■■■■"
0180 LINE$:="●●●●●●●●●●●●●●●●●●●●●●●●●●●●
     ●●●●●●●●●●●●"
0190 SECONDLINE$:="OOOOOOOOOOOOOOOOOOOO
     OOOOOOOOOOOOOO"
0200 SPACE$(1:40):="" // SET STRING SPACE
     EQUAL TO 40 BLANKS //
0210 CLEARSCREEN$:="■"
0220 HOME$:="■"
0230 UP$:="■"
0240 DOWN$:="■"
0250 //
0260 // INITIALIZE VARIABLES //
0270 //
0280 EOP:=FALSE // INITIALIZE END OF
     PROGRAM TO FALSE //
0290 CRT:=59468
0300 GRAPHICS:=12
0310 TYPEWRITER:=14
0320 //
0330 //
0340 // MAIN PROG  //
0350 //
0360 //
0370 EXEC SETBOARD
0380 AGAIN:
0390 WHILE EOP=FALSE DO
```

when you list this it is transformed into the following: FORJ:=1 to 10 DO. Here the shades of Pascal start to creep in. Notice the colon before the equals sign and the DO at the end. High level languages have always maintained that there is a difference between an equals statement and an assignment statement WHILE J=10 is considered different from J:=10 in the respect that the latter suggests that J will take on the value or be assigned the value 10. This is different from the previous statement that suggests that J is equivalent to 10. There is a fine line to this thinking but it is something that is easy to get used to. If you enter this code:

```
0010 FOR J = 1 TO 10
0020 FOR K = 1 TO 10
0030 FOR I = 1 TO 10
0040 PRINT I;
0050 NEXT I
0060 PRINT K;
0070 NEXT K
0080 PRINT J;
0090 NEXT J
0100 END
```

When viewing the listing you will see something like this:

```
0010 FOR J:=1 TO 10 DO
0020     FOR K:= 1 TO 10 DO
0030         FOR I:= 1 TO 10 DO
0040             PRINT I;
0050         NEXT I
0060         PRINT K;
0070     NEXT K
0080     PRINT J;
0090 NEXT J
0100 END
```

Notice that all the lines are indented to the proper nesting level of your code. This helps to view the code in a block like structure well known to most high level languages. Statements such as REPEAT, WHILE, UNTIL, ELIF and the IF-THEN-ELSE all contribute to the ability to write structured code. For example these two samples do exactly the same function. The main difference is that in the first example the control field is checked at the beginning as opposed to the end in the second. Note the two slashes used to denote comments as opposed to the familiar rem statement.

```
0010 I:=0
0020    WHILE    I<>=10
  //Control field//
0030     PRINT I,
```

```
0400    PRINT DOWN16$,
0410    INPUT "LINE TO CLEAR ? ": X
0420    IF X<0 THEN EOP:=TRUE
0430    EXEC CHECKNUMBER(X)
0440    IF NOT OK THEN GOTO AGAIN
0450    EXEC SETFLAG(X)
0460    EXEC CLEARLINE(X)
0470    EXEC WAIT
0480    EXEC CHECKLINE(X,FLAG(X))
0490 ENDWHILE
0500 PRINT CLEARSCREEN$
0510 POKE CRT,TYPEWRITER
0520 END
0530 //
0540 //
0550 // END OF MAIN PROG //
0560 // AND START OF PROCEDURES //
0570 //
0580 // NOTE THAT THE PROCEDURE CHECKLINE
     CALLS //
0590 // THE PROCEDURE CHECKFLAG WHICH IN
     TURN    //
0600 // MAY CALL ONE OF TWO OTHER
     PROCEDURES     //
0610 //
0620 //
0630 PROC CLEARLINE(LN)
0640    PRINT HOME$,
0650    FOR K:=1 TO LN DO PRINT DOWN$,
0660    PRINT SPACE$
0670    PRINT UP$,
0680 ENDPROC CLEARLINE
0690 //
0700 PROC FILLINE(X)
0710    PRINT HOME$,
0720    FOR J:=1 TO X DO PRINT DOWN$,
0730    PRINT J-1;LINE$
0740    PRINT UP$
0750 ENDPROC FILLINE
0760 //
0770 PROC FILLOTHERLINE(X)
0780    PRINT HOME$,
0790    FOR J:=1 TO X DO PRINT DOWN$,
0800    PRINT J-1;SECONDLINE$
0810    PRINT UP$
0820 ENDPROC FILLOTHERLINE
0830 //
0840 PROC SETBOARD
0850    POKE CRT,GRAPHICS
0860    PRINT CLEARSCREEN$
0870    FOR J:=1 TO 9 DO
0880       PRINT J;LINE$
0890    NEXT J
0900 ENDPROC SETBOARD
0910 //
0920 PROC WAIT
0930    FOR J:=1 TO 1500 DO
0940    NEXT J
```

```
0040      I:=I+1
0050 ENDWHILE
0010 I:=0
0020 REPEAT
0030      PRINT I,
0040      I:=I+1
0050  UNTIL  I□10
  //Control field//
```

Strains must be dimensioned before they may be referenced or used. The names of variables and strings may be up to 16 characters long as opposed to Basic's 2. The following shows the dimensions of two strings that will be differentiated from each other even though the first 15 characters are the same:

```
DIM ABCDEFGHIJKLMNOP$ OF 40
DIM ABCDEFGHIJKLMNOQ$ OF 40
```

The lines above dimensioned both strings to a maximum value of 40 characters. Strings may also be handled much like Atari or North Star Basic. For example:

```
0010 DIM A$ OF 30
0020 A$:="THIS IS THE NUMBER
  2 (ONE)"
0030 PRINT A$
0040 A$(23:3):="TWO"
0050 PRINT A$
0060 END
```

will produce the following results when the program is run.

```
THIS IS THE NUMBER 2 (ONE)
THIS IS THE NUMBER 2 (TWO)
```

Similarily in the above case a$(6:2) would be 18. This feature allows a lot more flexibility in string handling then the Left, Right, and Mid-String commands.

There is a Case statement similar to that of Pascal that is a more sophisticated way of conditionally executing code than the ON X GOTO command that we are all used to. A helpful feature about this Case structure is that it has an otherwise command that allows it to fall through to a default case if none of the conditions are met. It will take strings as well as numeric variables as the controlling factor.

```
0010 INPUT "ENTER A
  NUMBER FROM ONE TO
  FOUR": NUMBER$
0020 CASE NUMBER$ OF
0030     WHEN "ONE"
0040       PRINT 1
0050     WHEN "TWO"
0060       PRINT 2
0070     WHEN "THREE"
0080       PRINT 3
0090     WHEN "FOUR"
0100       PRINT 4
0110     OTHERWISE
0120       PRINT "WELL YOU
         TRIED ANYWAY
0130 ENDCASE
0140 END
```

```
0950 ENDPROC WAIT
0960 //
0970 PROC CHECKNUMBER(X)
0980   OK:=X<10 AND X>0
0990 ENDPROC CHECKNUMBER
1000 //
1010 PROC CHECKLINE(X,FLAG)
1020   CASE X OF
1030   WHEN 1
1040     EXEC CHECKFLAG(X,FLAG)
1050   WHEN 2
1060     EXEC CHECKFLAG(X,FLAG)
1070   WHEN 3
1080     EXEC CHECKFLAG(X,FLAG)
1090   WHEN 4
1100     EXEC CHECKFLAG(X,FLAG)
1110   WHEN 5
1120     EXEC CHECKFLAG(X,FLAG)
1130   WHEN 6
1140     EXEC CHECKFLAG(X,FLAG)
1150   WHEN 7
1160     EXEC CHECKFLAG(X,FLAG)
1170   WHEN 8
1180     EXEC CHECKFLAG(X,FLAG)
1190   WHEN 9
1200     EXEC CHECKFLAG(X,FLAG)
1210   ENDCASE
1220 ENDPROC CHECKLINE
1230 //
1240 PROC SETFLAG(X)
1250   IF FLAG(X)=0 THEN
1260     FLAG(X):=1
1270   ELSE
1280     FLAG(X):=0
1290   ENDIF
1300 ENDPROC SETFLAG
1310 //
1340 PROC CHECKFLAG(X,FLAG)
1350   IF FLAG=0 THEN
1360     EXEC FILLINE(X)
1370   ELSE
1380     EXEC FILLOTHERLINE(X)
1390   ENDIF
1400 ENDPROC CHECKFLAG
1410 //
```

A few more points that I would like to mention before we set into the demonstration program is that the GOTO statement is only valid to a label of 16 characters or less followed by a colon. EG:

```
0010 J:=0
0020 ENCORE:
0030 J:=J + 1
0040 PRINT J,
0050 GOTO ENCORE
0060 END
```

Subroutines are replaced by procedures and used by the command EXEC PROCEDURE NAME. The procedures may or may not use para-

meters as I will now show. Consider the following procedure:

```
0010 A:=5;B:=6;C:=8;D:=2
0020 EXEC ADDNUMBERS
0030 END
0040 // Start of procedure to add
     numbers//
0050 PROC ADDNUMBERS
0060 PRINT A + B
0070 ENDPROC
0080 //
```

The result will be 11 since all four variables are global or public; that is they are known and can be used by the whole program. This is different from the next example that incorporates local variables only known to the procedure:

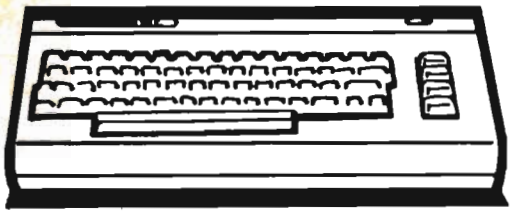
```
0010 A:=5;B=6;C=8;D:=2
0020 EXEC ADDNUMBERS(C,D)
     // C and D are passed //
0030 END
0040 // Start of procedure to add
     numbers //
0050 PROC ADDNUMBERS(A,B)
0060     PRINT A + B
0070 END PROC
0080 //
```

Have you all guessed what the answer will be? If you said 11 again I can't blame you, but the answer is 10. The variables A and B in the procedure have nothing to do with the global variables declared at the beginning. Since they are only known to that procedure as mentioned earlier A in the procedure takes on the value of C and B takes on the value of D giving us our answer. If I had used local variables such as J or K you may have figured it out easier, but what fun would that have been? By utilizing and and testing procedures before you write your main driving program you can be fairly sure that your program will work on the first runs. Another reason for this is that unlike basic the syntax is checked on input so you can only run time errors.

Reffering back to a statement that Jim Butterfield made in his article on COMAL about there not being a SYS command for an exit to BASIC, well he should be happy to know that the later versions did include it. This also means that you can use machine language subroutines just as in Basic. As a matter of fact there are a lot of useful commands not available in Basic. Here is a compilation of them. I won't go into their functions but I believe most of them are obvious.

I have included a short sample program that demonstrates further the concepts that I have been discussing. It simply displays 9 rows of black dots and when asked to pick a row it disappears and then returns as white dots. It acts like a flip-flop since a flag is set and when that line is chosen again it reverts to its original colour. The only aspects of COMAL that it uses that I haven't mentioned is the boolean functions (TRUE and FALSE) and the way it is used in the procedure CHECKNUMBER. CHECKNUMBER may be referred to as a typed procedure since it returns a value to us; in this case the boolean value of OK that is set to true or false by checking the limits set for x. Most of the other procedures are considered untyped since they do not return any value to us but just execute the function they are meant to do. The mainline program is right after the variable and string declarations, followed by the procedures. I believe that the procedure names clearly define what they do so there should be no need to describe them as one usually must do in BASIC. I hope you enjoy the program and will modify the code to get the feel of this interesting language. In a sequel to this article I will go further into the aspects of COMAL and get into file handling, etc. Have fun and let's hear from some more of the COMAL users out there.

ABS	AND	ATN	AUTO	BASIC
CASE	CHAIN	CHR	CLOSE	CLOSED
CON	COS	DATA	DEBUG	DEL
DIM	DIV	DO	ELIF	ELSE
END	ENDCASE	ENDIF	ENDPROC	ENDWHILE
ENTER	EOD	EOF	ESC	EXEC
EXP	FALSE	FOR	GOTO	IF
IN	INPUT	INT	LABEL	LEN
LET	LIST	LOAD	LOG	MOD
NEW	NEXT	NOT	OF	OPEN
OR	ORD	OTHERWISE	OUTPUT	PRINT
PROC	READ	REF	REM	RENUM
REPEAT	RESTORE	RND	RUN	SAVE
SELECT	SGN	SIN	SIZE	SPC
SQR	STATUS	STEP	STOP	TAB
TAN	THEN	TIME	TO	TRAP
TRUE	UNTIL	USING	WHEN	WHILE
WHILE	ZONE	EDIT	CAT	APPEND
RANDOM	UNIT	FILE	POKE	PEEK
SYS				



VIC 20

Commander would like to apologize for an error in our December 1982, Premier Issue. We neglected to print the listing in the Enterprise article. Due to this error we are republishing the article "Enterprise" with its listing. Again, we would like to apologize for this error.
The Editor

Enterprise

by Tim Parker
Ontario, Canada

One of the most popular types of computer games is the genre patterned on the television series **Star Trek**. Versions exist for almost every type of programmable machine, ranging from hand held calculators to super computers. These games generally fit perfectly into the small home computer field, as the home computer can employ graphics and sound, along with cursor manipulations that are seldom found on larger systems. The trade-off, of course, is memory.

The VIC series provides an interesting challenge for a programmer who wishes to creat a TREK game. The small screen size of the VIC-20 is ideally suited to a game of this sort, allowing graphic block positioning. The 64, with its larger screen (and memory), can make use of the extra space for more involved displays. Although a version can be written to fit in an unexpanded VIC-20 with a 3.5k of memory, the limitations are quite severe. However, with an extra 3k of memory, a fairly good program can be achieved.

This article deals with the game ENTERPRISE, which fits nicely in the 6.5k of an expanded VIC-20. By changing a few memory screen pointers, the program will also run

```
10 V=36878:S2=V-3:GOSUB6000
20 PRINT"{CLEAR}":POKE36879,141
30 X=RND(-TI):V=36878:S4=V-1:S3=V-2:S2=
   V-3:S1=V-4
40 C=99:GOSUB1000
50 DEFFNA(X)=INT(RND(1)*X)+1
60 DEFFNL(X)=(7724+FNA(10)*22)+FNA(10)
70 B(FNA(9))=1:BT=1
80 FORZ=1TOFNA(9):K(FNA(9))=FNA(4):NEXT
   :Q=FNA(9)
90 FORZ=1TO9:KT=KT+K(Z):NEXT:IFKT<11THE
   NCLR:GOTO30
300 GOSUB10000
310 FORA=1TOFNA(7):POKEFNL(1),42:NEXT
400 IFK(Q)>0THENFORA=1TOK(Q):POKEFNL(1)
    ,11:NEXT
410 IFB(Q)>0THENPOKEFNL(1),2
420 LE=FNL(1):IFPEEK(LE)<>32THEN420
500 GOSUB10000:GOSUB11000
510 POKELE,5
520 IFS<20THENGOSUB5000
600 IFE<00RC<0THEN9000
610 PRINT"{BLUE}C=COMPUTER":PRINT"F=FIRE
    WEAPON":PRINT"M=MOVE SHIP"
620 PRINT"{RVS}COMMAND"
660 GETA$:IFA$=""THEN660
670 IFA$="M"THEN2000
680 IFA$="F"THEN3000
690 IFA$="C"THEN4000
700 GOTO660
1000 E=99:S=99:SC=99:P=99:T=99:I=99:W=9
```

with more memory, or using the 64. By trimming out the "flash", a pared down unexpanded version can be used. (It can be modified to run on PETs, and other computers by changing color controls, and in some cases, cursor control characters. Memory location pointers will also require changing, but this is primarily a text based game, so conversion will be fairly simple.)

In order to utilize memory in the most efficient manner, and to enable fast program execution, screen displays are generated using POKE commands. The primary saving here is in not having to use a matrix for the screen display. (A 10x10 matrix can consume over half a kilobyte of memory.)

By cutting down on the number of text statements, further memory saving is achieved. While the game may lack some of the features of the 40k (and up) Trek versions for computers such as the Apple, it includes many of the extras that are most interesting.

Generation of most of the game factors is controlled by random number generators. This is initialed by reference to the built in clock function.

Color is used to separate the screen into logical sections, and to add a little bit of variety to the display. Sound programming in this game was constructed for the VIC-20's sound registers. These can be converted to the 64's more involved sound device by reference to the manual. Alternatively, sound can be omitted, although the saving in terms of memory space is minimal.

When RUN, the screen display is shown in the upper left. A matrix grid is labelled by the numbers 0 to 9 on the top and left axes. These correspond to the X and Y coordinates respectively. The video display uses an asterisk to represent a star, a "K" for the Klingons, and an "E" for the Enterprise. If a starbase exists, it is represented by a "B". (If extra memory is available, custom characters can be generated for these items.)

The upper right shows the Enterprise's status at all times. The Universe of the game is a 3x3 matrix, giving nine quadrants. The current

```
9:RETURN
2000 GOSUB11000
2010 PRINT"{RVS}W{OFF}ARP OR {RVS}I{OFF}
     MPULSE?"
2020 GETA$:IFA$=""THEN2020
2030 IFA$="I"ANDI>0THEN2200
2040 IFA$="W"ANDW>0THEN2070
2050 GOTO500
2070 GOSUB11000
2080 PRINT"ENTER DESTINATION"
2100 PRINT"QUADRANT NUMBER":PRINT"(1-9)
     ";
2110 GETA$:IFA$=""THEN2110
2120 B=VAL(A$):IFB<1ORB>9THEN2110
2130 PRINTB:GOSUB7000
2140 IFB=QTHEN500
2150 POKEV,10:FORZZ=1TO9:POKES1,220:FOR
     ZX=1TO120:NEXT
2160 POKES1,0:FORZX=1TO30:NEXTZX,ZZ
2170 POKEV,0:POKES1,0
2180 Q=B:W=W-FNA(9):PRINT"{CLEAR}":IFW<0T
     HENW=0
2190 GOTO300
2200 GOSUB12000
2210 L1=7747+Y*22+X:I=I-FNA(9)
2220 IFI<0THENI=0
2230 IFPEEK(L1)=2THENGOSUB1000:BT=0:GOT
02300
2240 IFPEEK(L1)<>32 THEN500
2300 GOSUB7000
2310 POKES4,175:POKEV,10:FORZX=1TO600:N
     EXT
2330 FORZZ=10TO0STEP-1:POKEV,ZZ
2340 FORZX=1TO100:NEXTZX,ZZ:POKES4,0
2350 POKELE,32:LE=L1
2360 POKELE,5:GOTO500
3000 GOSUB11000
3010 PRINT"{RVS}P{OFF}HASER OR {RVS}T{OFF}
     ORPED 0?";
3020 GETA$:IFA$=""THEN3020
3030 IFA$="P"ANDP>0THENP=P-FNA(8):WE=1:
     GOTO3400
3040 IFA$="T"ANDT>0THENT=T-FNA(8):WE=2:
     GOTO3400
3050 GOTO500
3400 IFT<0THENT=0
3410 IFP<0THENP=0
3420 GOSUB10000:GOSUB12000
3430 O=7747+Y*22+X
3440 IFPEEK(O)<>11THEN500
3450 PRINT"{DOWN}{BLACK}{RVS}FIRING":
     C=C-FNA(2)
3460 IFWE=1THENGOSUB3800
3470 IFWE=2THENGOSUB3900
3480 IFFNA(9)<5THEN3600
3490 POKEO,32:K(Q)=K(Q)-1:KT=KT-1
3500 PRINT"{RVS}{WHITE}{DOWN}KLINGON
     DESTROYED"
```

quadrant is shown at the top of the status display.

Below the quadrant, the ship's functions are all represented by efficiency ratings from 0 to 99. They are ENGY (energy), SHLD (shield), COMP (computer), SCAN (scanner), PHSR (phaser), TORP (photon torpedoes), IMPL (impulse engines) and WARP (warp engines). After any action, these ratings are updated.

On each turn, there are three primary commands that can be executed: COMPUTER, MOVE or FIRE. MOVE allows either impulse movement (within the current quadrant) or warp movement (to another quadrant) as long as the respective ratings are not zero.

FIRE will target either photon torpedoes or phasers on a specified coordinate. As the firing is computer controlled, there is little dependance on distance to target within a quadrant.

COMPUTER allows several different functions to be carried out. A scanner will give the number of enemy ships and starbases remaining in the universe. Self destruct does exactly that. (To be used only in the face of overwhelming odds, or by masochists.) The SET command allows the ratings of any of the status functions to be raised or lowered. (Except computer and scanner—these are considered to be unfixable.) If a function is lowered, the excess units are diverted to ENGY (think of them as auxiliary batteries) up to a maximum of 99. Note that any energy above 99 is lost! If a function is raised, the required units are diverted from ENGY, to a minimum of zero.

Strategy is a matter of personal preferences. At each quadrant where there are enemy ships, they will take shots at the Enterprise on any FIRE or MOVE command. (As computer commands are considered to be carried out almost simultaneously, the enemy does not fire at you when the computer is used.)

There is the possibility of the enemy calling in reinforcements if the battle goes poorly for them, or if they sense victory.

All hits by the enemy are deducted from the shield rating. If this drops too

```
3510 GOSUB3700
3600 IFKT=0THEN8000
3610 GOSUB7000:GOTO500
3700 VP=VP+25:TN=130
3710 FORZZ=15TO0STEP-1
3720 POKEV,ZZ:POKES4,TN
3730 FORZX=1TO230:NEXT:NEXT:POKES4,0:
     RETURN
3740 POKES4,0:RETURN
3800 FORZZ=12TO0STEP-2.5:POKEV,ZZ
3810 FORZX=220TO225
3820 FORZY=225TO220STEP-1
3830 POKES2,ZX:POKES3,ZY
3840 NEXTZY,ZX
3850 FORZX=220TO225
3860 FORZY=225TO220STEP-1
3870 POKES2,ZY:POKES3,ZX
3880 NEXTZY,ZX
3890 NEXT:POKES2,0:POKES3,0:RETURN
3900 TN=210
3910 FORZZ=15TO0STEP-1
3920 POKEV,ZZ:POKES4,TN:POKES1,TN
3930 FORZX=1TO130:NEXT
3940 TN=TN-.5:NEXT
3950 POKES4,0:POKES1,0:RETURN
4000 GOSUB11000:IFC=0THEN500
4005 C=C-FNA(6):IFC<0THENC=0
4010 FORZZ=1TO4:POKEV,10:POKES3,200
4015 FORZX=1TO50:NEXTZX:POKES3,0
4020 FORZX=1TO30:NEXTZX,ZZ
4025 POKES3,0:POKEV,0
4030 PRINT"1=SCANNER":PRINT"2=SET
     SHIELDS"
4040 PRINT"3=SET PHASER"
4050 PRINT"4=SET TORPEDO":PRINT"5=SET
     IMPULSE ENG"
4060 PRINT"6=SET WARP ENG":PRINT"7=SELF
     DESTRUCT"
4070 GETA$:IFA$=""THEN4070
4080 A=VAL(A$):IFA<1ORA>7THEN4070
4090 GOSUB11000:ON AGOTO4100,4200,4300,
4400,4500,4600,4700
4100 SC=SC-FNA(8)
4110 IFSC<0THENSC=0:GOTO500
4120 PRINT"THERE ARE";KT;"KLINGONS"
4130 IFBT=1THENPRINT"AND 1 BASE LEFT.":
     GOTO4150
4140 PRINT"AND 0 BASES LEFT."
4150 POKEV,10:FORZZ=1TO90
4160 POKES3,FNA(100)+128
4170 FORZX=1TO10:NEXTZX,ZZ
4180 POKEV,0:POKES3,0
4190 GOTO500
4200 PRINT"SHIELD RATING":INPUT"(0-99)"
     ;A
4210 IFA<00RA>99THEN500
4230 IFA<STHEN4260
```

low, a "red alert" warning is issued. At this point, reinforcement of the shields is highly advisable. If the shield rating drops to zero, enemy damage is deducted from ENGY, with the further possibility of internal damage to the other functions. If the computer drops to zero, or both shields and energy are zero, then the game is over, and the enemy is triumphant.

Computer rating drops when a weapon is fired, as it acts as a targeting device. The scanner drops when it is used. (If the scanner rating is zero, the scanner is inoperational.) If weapons or engines fall to zero, they are useless until the computer is used to increase their rating.

Impulse engines are of little use in combat, as distance to enemy is not a factor in hit probability. The impulse engines are primarily used to manoeuvre to starbases for refuelling. A wise captain uses the impulse engines as a battery backup for energy and shield.

If a starbase is landed on, the ratings of the ship all increase to maximum, except for the computer. Using a starbase to refuel will affect your performance rating at the end of the game, when a numerical score will be given that reflects the number of enemy destroyed, and the Enterprise's status at game conclusion.

The program is designed in a modular method, allowing quick modification and reference to any sections. Initiallization and enemy distribution is established in the first few lines. The control loop is lines 500-700, which direct further branching to any relevant subroutines.

The program is broken down as follows:

70-90 Klingon distribution and base location
300-500 POKEing of K, E and B onto grid
500-700 Control loop
1000 Refuelling

Most of the variable's functions are easily identified from their context. Line length has been limited in most cases to one statement per line to simplify programming and debugging. The major exception is

```
4240 E=INT(E-A+S):S=A:IFE<0THENS=S+E:E=
     0
4250 GOTO500
4260 E=INT(E+S-A):S=A:IFE>99THENE=99
4270 GOTO500
4300 PRINT"PHASER RATING":INPUT"(0-99)"
     ;A
4310 :IFA<00RA>99THEN500
4330 IFA<PTHEN4360
4340 E=INT(E-A+P):P=A:IFE<0THENS=S+E:E=
     0
4350 GOTO500
4360 E=INT(E+P-A):P=A:IFE>99THENE=99
4370 GOTO500
4400 PRINT"TORPEDO RATING":INPUT"(0-99)
     ";A
4410 IFA<00RA>99THEN500
4430 IFA<TTHEN4460
4440 E=INT(E-A+T):T=A:IFE<0THENT=T+E:E=
     0
4450 GOTO500
4460 E=INT(E+T-A):T=A:IFE>99THENE=99
4470 GOTO500
4500 PRINT"IMPULSE RATING":INPUT"(0-99)
     ";A
4510 IFA<00RA>99THEN500
4530 IFA<ITHEN4560
4540 E=INT(E-A+I):I=A:IFE<0THENI=I+E:E=
     0
4550 GOTO500
4560 E=INT(E+I-A):I=A:IFE>99THENE=99
4570 GOTO500
4600 PRINT"WARP ENG RATING":INPUT"(0-99)
     ";A
4610 IFA<00RA>99THEN500
4630 IFA<WTHEN4660
4640 E=INT(E-A+W):W=A:IFE<0THENW=W+E:E=
     0
4650 GOTO500
4660 E=INT(E+W-A):W=A:IFE>99THENE=99
4670 GOTO500
4700 PRINT" {RVS}SELF DESTRUCT ACTIVE"
4710 POKEV,12
4720 FORZZ=15TO0STEP-1:POKES4,150+ZZ*3
4730 POKE36879,FNA(200)+8
4740 POKEV,ZZ:FORZX=1TO150:NEXTZX,ZZ
4750 POKE36879,141:PRINT"{CLEAR}"
4760 GOSUB10000:POKES4,0
4770 PRINT"{2DOWN}  DESTRUCT COMPLETED"
4780 :PRINT"{DOWN}      SCORE=";INT(VP/2):
     GOTO8200
5000 FORZQ=1TO4
5010 PRINT"{HOME}{RVS}{RED} CONDITION RED
     ";
5020 GOSUB5500
5030 PRINT"{HOME}                        ";
5040 GOSUB5500:NEXT
```

the IF...THEN statement. All statements after the IF...THEN are executed if the conditional is true, allowing the avoidance of multiple IFs or subloops. For byte savers, the number of statements per line can be increased, thereby saving several bytes for each line number omitted.

The difficulty of the game can be changed in line 90, where KT refers to the total number of Klingons. Increasing the maximum from 11 will increase the difficulty. The hit probability is given in line 3480. Decreasing the number in the FNA (X) statement will decrease the difficulty.

As the variables are in this listing, the game is relatively easy to win once a strategy has been determined as effective. There is however always the possibility of a few damaging hits from the Klingons that totally ruin shields. That's when things get very interesting!

When using ENTERPRISE on a VIC-20 with more memory, the screen pokes have to be changed. Line 60 defines the location of the memory locations for these screen pokes. As set with the value 7724, the program will run on unaltered and 3k expanded VICs. (The memory map actually starts at 7680; the extra 44 positions the cursor at the start of the display grid. Color starts at 37888.) With 8k or more, substitute the value 4140. (The screen memory starts at 4096, and the color memory at 37888.)

2000-2999	Movement (Warp and Impulse)
3000-3999	Fire control
4000-4999	Computer functions
5000-5999	Red Alert loop
6000-6999	Introductory loop
7000-7599	Klingon fire loop
7600-7999	Reinforcement loop
8000-8999	Victory loop
9000-9999	Destruction loop
10000-10999	Status and screen grid
11000-11999	Cursor positioning
12000-12999	Coordinate loop

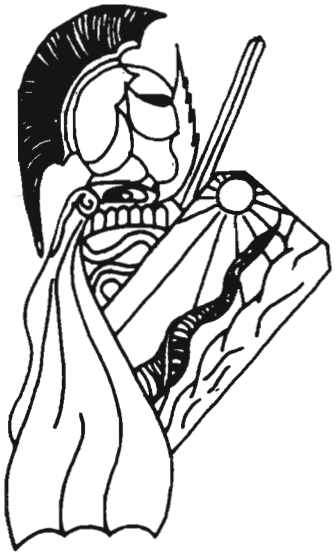
With extra memory, the program can be expanded in ways that are limited only by the programmer's imagination. The 3x3 universe can be expanded, as can the quadrants.

```
5050 PRINT"{HOME}{RVS}{RED} CONDITION RED
     "
5060 PRINT"{13DOWN}"
5070 RETURN
5500 POKEV,10
5510 FORZA=180TO226STEP2:POKES3,ZA
5520 FORZS=1TO7:NEXTZS,ZA
5530 POKES3,0:FORZD=1TO10:NEXT
5540 POKEV,0:RETURN
6000 POKE36879,25
6010 PRINT"{CLEAR}{11DOWN}{BLACK}
     ENTERPRISE"
6020 POKEV,12
6030 READA,B
6040 IFB=-1THEN6200
6050 FORZZ=1TOB:POKES2,A
6060 NEXT:GOTO6030
6200 FORZZ=12TO0STEP-1
6210 POKEV,ZZ
6220 FORZX=1TO50:NEXTZX,ZZ
6230 POKES2,0:FORZZ=1TO20:PRINT
6240 FORZX=1TO40:NEXTZX,ZZ:RETURN
6800 DATA135,150,163,80,187,220,183,60,
     163,60
6810 DATA147,60,175,70,195,180,-1,-1
7000 IFK(Q)<1THENRETURN
7010 GOSUB11000:D=0
7020 FORX=1TOK(Q):D=D+FNA(8):NEXT
7025 PRINT"{RVS}{BLUE}KLINGONS FIRING":
     PRINT"DAMAGE=";D:S=S-D:VP=VP-D/2
7030 IFS<0THENE=E+S:S=0:PRINT"{WHITE}SHIE
     LDS DOWN":PRINT"{RVS}INTERNAL DAMAGE"
7040 IFS=0ANDFNA(4)=1THENC=C-FNA(20):IF
     C<0THENC=0
7050 IFS=0ANDFNA(4)=1THENSC=SC-FNA(20):
     IFSC<0THENSC=0
7060 IFS=0ANDFNA(4)=1THENP=P-FNA(20):IF
     P<0THENP=0
7070 IFS=0ANDFNA(4)=1THENT=T-FNA(20):IF
     T<0THENT=0
7080 IFS=0ANDFNA(4)=1THENI=I-FNA(20):IF
     I<0THENI=0
7090 IFS=0ANDFNA(4)=1THENW=W-FNA(20):IF
     W<0THENW=0
7100 POKEV,10:FORZZ=1TO15
7110 FORZX=200TO220+ZZ:POKES2,ZX:NEXTZX
     ,ZZ
7120 POKES2,0:POKEV,0
7400 IFFNA(4)=1THENGOSUB7600
7410 RETURN
7600 GOSUB11000:PRINT"KLINGONS RADIOING
     FOR":PRINT"HELP...."
7610 POKEV,10:POKES4,230:FORZX=1TO2500:
     NEXT:POKES4,0:POKEV,0
7620 IFFNA(3)>1THENRETURN
7630 PRINT"KLINGON WARPING INTO":PRINT"
     SECTOR"
```

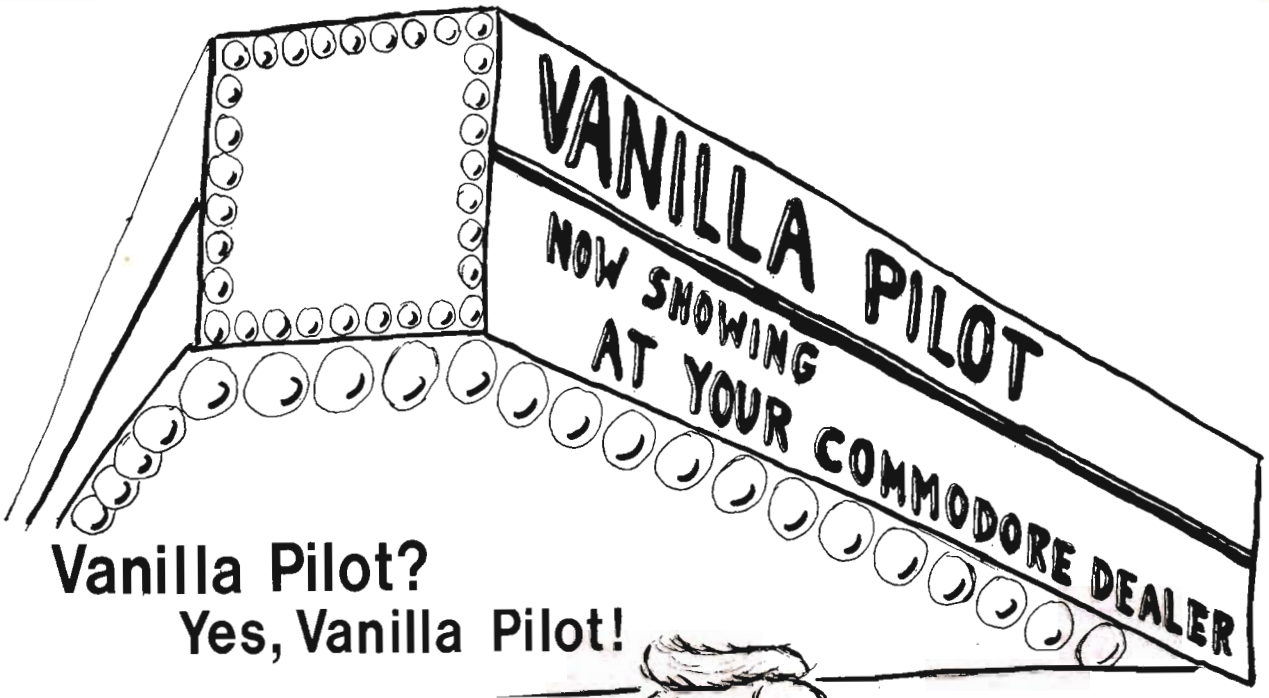
Different enemy classes can be incorporated, and a more sophisticated targeting and hit algorithm can be used. The "fog of war" can be imitated by adding limited intelligence as the scanner and computer ratings drop. Mutinies, shuttlecraft, drones and many other features can add to the complication of the game.

On the output side, the sound can be altered (especially with the 64's exceptional sound generators), and use can be made of multiple screens. Animation is possible for the ambitious.

Although several commercial versions of Trek games are now available, this program was designed from scratch to conform to the author's concepts. Comparison with the commercial products has shown areas where it is deficient, and where it excells. Placing the game in a magazine such as this ensures that people will have a choice, and do not have to pay inflated software prices. Enjoy, and may the Klingons all be cowards!



```
7640 POKEV,10:FORZZ=230TO150STEP-1:POKE
     S2,ZZ:FORZX=1TO20:NEXTZX,ZZ:POKEV,0:
     POKES2,0
7650 ZZ=FNL(1):IFPEEK(ZZ)<>32THEN7650
7660 POKEZZ,11:K(Q)=K(Q)+1:KT=KT+1:
     RETURN
8000 GOSUB11000:PRINT
     "YOU WIN"
8010 IFBT=1THENVP=VP+(VP*.25)
8020 IFC=0THENVP=VP-(VP*.1)
8030 IFS=0THENVP=VP-(VP*.2)
8090 VP=INT(VP)
8100 PRINT"{2DOWN}   SCORE=";VP
8200 PRINT"{2DOWN}{RVS}ANOTHER GAME?"
8210 GETA$:IFA$=""THEN8210
8320 IFA$="Y"THENRUN
8400 PRINT"{CLEAR}":END
9000 GOSUB11000:PRINT"{RVS} SHIELDS
     DESTROYED"
9010 PRINT"LIFE SUPPORT KNOCKED":PRINT"
     OUT...YOU LOSE."
9020 PRINT"      SCORE=0!":GOTO8200
10000 PRINT"{HOME}{2DOWN}{BLACK} 0123456789"
10010 PRINT"0":PRINT"1":PRINT"2":PRINT"
      3":PRINT"4"
10020 PRINT"5":PRINT"6":PRINT"7":PRINT"
      8":PRINT"9"
10100 PRINT"{HOME}{DOWN}"
10110 PRINTSPC(12)"{RVS}{WHITE}QUAD= ";Q
10120 PRINT:PRINTSPC(12)"{YELLOW}ENGY=
      {3LEFT}";E
10130 PRINTSPC(12)"SHLD=   {3LEFT}";S
10140 PRINTSPC(12)"{CYAN}COMP=   {3LEFT}";C
10150 PRINTSPC(12)"SCAN=   {3LEFT}";SC
10160 PRINTSPC(12)"{BLUE}PHSR=   {3LEFT}";P
10170 PRINTSPC(12)"TORP=   {3LEFT}";T
10180 PRINTSPC(12)"{RED}IMPL=   {3LEFT}";I
10190 PRINTSPC(12)"WARP=   {3LEFT}";W
10200 RETURN
11000 PRINT"{HOME}{12DOWN}"
11010 FORZ=1TO9:PRINT"
      ":NEXT
11020 PRINT"{8UP}":RETURN
12000 GOSUB11000:PRINT"ENTER X
      COORDINATE";
12010 GETA$:IFA$=""THEN12010
12020 X=VAL(A$):IFX<00RX>9THEN12010
12030 PRINTX
12040 PRINT"ENTER Y COORDINATE";
12050 GETA$:IFA$=""THEN12050
12060 Y=VAL(A$):IFY<00RY>9THEN12050
12065 PRINTY
12070 RETURN
```



Vanilla Pilot?
Yes, Vanilla Pilot!

What is Vanilla Pilot?

Vanilla Pilot is a full-featured pilot language interpreter including TURTLE GRAPHICS for the PET or CBM 4000, 8000, 9000 and CBM-64 series computers.

At last! A Pilot interpreter for the Commodore computers. This Pilot includes some powerful extensions to the screen editor of the computer. Things like FIND/CHANGE, TRACE and DUMP enhance the programming environment.

The TURTLE has a very powerful set of graphics commands. You can set the Turtle's DIRECTION and turn him LEFT or RIGHT. The pen he carries can be set to any of the 16 colors in the CBM-64. He can DRAW or ERASE a Line.

What else? Vanilla Pilot is all this and much, much more. In fact, we can't tell you about all of the features of the language in this small ad. So rush down to your local Commodore computer dealer and ask him to show you Vanilla Pilot in action. Be sure to take the $2.00 discount coupon.

Hurry, you have only a short time to redeem your coupon. So use it now!

Tamarack Software
Darby, MT. 59829

SAVE $2.00

VANILLA PILOT

Retailer: Send the redeemed coupons to Tamarack Software, Darby, MT 59829. We will pay $2. plus $.35 handling for the redemption of these coupons. If requested, invoices showing sufficient purchase of Vanilla Pilot must be submitted. Coupons submitted to us more than 30 days after the expiration date will not be honored.

Expires April 15, 1983.

RAVINGS OF A MADMAN

by Tim Parker
Ontario, Canada

When all is said and done, most people buy their computers for playing games. Certainly this reason is not the primary justification at purchase time. The computer is bought to balance cheque books, teach youngsters the miracles of future gadgets, or allow a painless entry in today's computerized marketplace.

All of that, of course, is fine, but totally irrelevant. Most computers are bought for game playing. (Here the distinction should be made between the "home computer" such as the VICs, Apples, etc., and the true small computer. The latter usually does not have a color screen, sound capabilities, or graphics. It runs CP/M or a similar operating system, and is intended for true computational or computer assisted work.)

Luckily, it is easy to justify the cost when a VIC-20 setup can be obtained for the same cost as a home video system. And the game quality is comparable, despite what many people feel. The VIC-20 now has quite a large range of preprogrammed software available for it, while the newer VIC-64 is still in its early stages.

So what are good games for the VIC-20? It depends on what you like! That seemingly useless advice has some truth behind it. An arcade player would certainly not want to sit down to a high level chess or Othello game, while and adventure game player may disdain the shoot-em-up video games that are the rage.

What you want decides where you go to get it. The cartridges brought out by Commodore in the initial release were not the best available by any stretch of the imagination. The traditional invaders game ("Avengers") is a classic, and deserves a place in any videophile's collection, but it is by no means 'state of the art.' The rest of Commodore's cartridges are interesting, but lack distinction when compared to what else is available.

A very good line of video game cartridges and programming aids is available from United Microware Industries. Their games range from "Amok" (a "Berserk" look-alike), through "Spiders of Mars", an addictive game based loosely on "Defender", and several others. UMI was the first company to get in on the VIC-20 with a good line of games. (Their address is listed at the end of this column.)

One company that has developed over the past six months or so is Protecto Enterprises, which acts as a retailer of many items for the VIC-20. They cover the major items on UMI's list, as well as many other companies. Their prices are usually very competitive.

Adventure gamers can turn to Scott Adams' Adventure International, who have adapted a few of their more popular games onto cartridges for the VIC. Commodore also has new adventure cartridges released.

Chess, and other "high-brow" games are still in a low end of the development scale. The release of Sargon chess was good, as it plays a well reasoned game, but surely that does not represent the limit of the VIC's capabilities. Several other companies have released non-video games for the VIC, although only time will tell if they stand up to the market.

Perhaps the cheapest way to get games for the VIC is through programming them yourself. Many books are available, usually with titles such as "Fifty Thousand Games For Your Home Computer". The vast majority of these books are a disappointment. Only a vew versions of Hangman need to be seen before the novelty wears thin. A few good books are available however. Two of my favorite compendia of games are published by Workman Publishing. Edited by David Ahl, the books ("Basic Computer Games" and "More Basic Computer Games") contain a potpourri of different programs. Most are not too exciting, but a few are worth the time to key them in. These two are definite recommendations for the well-stocked book shelf.

Finally, the other major source of programs is the computer magazine. There is quite a large variety available now, although some cater only to certain specialties. However, there are two or three that regularly publish interesting games (including this erudite journal) that go beyond the edge of boredom. Be choosy, and careful, and many wasted hours can be avoided.

Above all, enjoy yourself! A few minutes sitting in front of a television or monitor can be relaxing, if approached correctly. The major problem is that many people get uptight about the whole affair, and treat each game as a live or die situation. Remember it's only a game!

There are a few quick shortcuts available to the VIC-20 programmer that allows keying in of often used commands. The best known example is that of performing a LOAD followed by a RUN: hold down the shift key and press RUN/STOP. This initiates the cassette load sequence. When the program is LOADed, a RUN is automatically supplied.

The semi-warm boot is also fairly useful. (Time for a couple of definitions. A "boot" is when the computer control system is reloaded fresh into memory, or accessed by a ROM. A "cold boot" occurs when the computer is turned off, then on again. This initializes everything. A "warm boot" is used in more advanced computer systems to refer to reloading the operating system without turning the machine off. This usually leaves memory intact.)

As the VICs do not have an operating system as such, a warm boot will be the same as the reset command (holding down the RUN/STOP key and hitting RESTORE) which resets some memory pointers but doesn't harm the memory contents. The semi-warm boot I refer to above is a term used for performing a cold boot without physically turning the machine off. This is accomplished with a SYS call (machine language call) to the processor.

The instruction SYS 64802 will perform the same task as a cold boot, without having to touch the on/off switch.

There are two ways to perform a RUN. One is a SYS call (SYS 50830). The other is a three key sequence. Hold down the left shift key, press 2 (to give a quotation mark) then RUN/STOP. The screen will give a little wiggle, then a run will start. This is not as useful a trick as might be imagined. Typing R-U-N is not all that demanding, but this does allow one-handed RUNs when the other hand is busy.

There are a few more tricks that are known, but of limited value. Next column, I'll mention a few that act as a security system for your programs, preventing SAVEing or LISTing, and aborting the RUN/STOP key. Stay tuned.

Finally, now that the VIC modems are available almost everywhere with few supply problems, it's worth taking a look. A modem (it stands for MOdulator-DEModulator) allows connection of the computer to a telephone line, and subsequent connection to either another computer, or one of the bulletin board (or similar) services available throughout the continent. The cost is very reasonable, and the number of program that can be added to your library is quite large, depending on which service you get. Some users groups are offering modem facilities.

One problem with the VIC-20 is its screen size. Don't expect the VIC to act as a high priced terminal. It can't. Most terminals have a 24x80 screen minimum (some allow 130 columns). The VIC can't compete with that. Is it a problem? Not really. The VIC probably won't be used for the same purposes as the larger terminals, so it is adequate.

For those who want the larger screen size, there are many video screen expanders available now that give forty, or even eighty columns on the VIC. But I'll leave that for next month.

United Microware Industries Inc., 3503 Temple Ave., Suite D, Pomona, CA 91768. (714) 594-1351.

Protecto Enterprises, Box 550, Barrington, Illinois 60010. (312) 382-5244.

GOBBLE!

by Tim Parker
Ontario, Canada

Gobble! is relatively easy to play, but is not such a simplistic game that interest quickly fades. It is a game ideally suited to the screen size of a VIC-20, although it can easily be modified to the larger size of the VIC-64, PET, or other computers. It fits inside the memory of an unexpanded VIC-20.

The game is based upon a maze generation program that draws a unique maze (i.e. has one entrance and one exit). Several people have developed the algorithms required for this purpose, including David Matuszek (*BYTE*, December 1981, Vol 6 No 12) and Charles Bond (*COMPUTE!*, December 1981, Vol 3 No 12). The latter has been adapted for use in this program, although most variables had to be changed to accomodate the screen size. The details of the maze generator segment will be left for the reader to investigate.

The object of the game, simply stated, is to "eat" as many dots as possible with your character (a ball) while avoiding the enemy (a square) that runs the maze at the same time as you, albeit slower. For each dot you eat, you get one point. For each dot the enemy eats, you lose one point.

In the early levels of the game, there are several cross connections provided to make escaping the enemy easy. As the levels are completed, the number of connections becomes fewer, and the game subsequently more difficult.

Bonus points are awarded on completion of the fourth, ninth, and fourteenth rounds (assuming you get that far). A player may exit the maze through the hole in the bottom at any time, whether all the dots are eaten or not. The next level will then be generated.

The version printed here is intended for keyboard control. Modification of joystick is easy, and the details are

JOYSTICK MODIFICATIONS

Add the following lines:

```
9000 DD=37154: P1=37151: P2=37152
9010 POKE DD, 127: P=PEEK(2)AND128: jO=-(P=O)
9020 POKE DD,255: P=PEEK(P1)
9030 J1=([PAND8]=-0): J2=-([PAND16]=0): J3=-
     ([PAND4]=0): RETURN
90   GOSUB 9000
100  IFJQANDPEEK(L+1)<> 160THENPOKEL,32:L=L+1
     :GOSUB600: POKEL,81
110  IFJ2ANDPEEK(L-1)<> 160THENPOKEL,32:L=L-1
     :GOSUB600: POKEL,81
120  IFJ1ANDPEEK(L+22)<>160THENPOKEL,32:L=L+22
     :GOSUB600: POKEL,81
130  IFJ3ANDPEEK(L-22)<> 160THENPOKEL,32:L+L-22
     :GOSUB600: POKEL,81
```

```
20 POKE36879,25:V=36878:SO=36875:PT=0:D
L=0
30 POKEV,10
40 GOSUB6000
50 L=INT(RND(-TI)):DEFFNA(X)=INT(RND(1)
*X)+1
60 DL=DL+1:IFDL=50RDL=100RDL=15THENGOSU
B5000
65 GOSUB1000
70 L=7734:POKEL,81:POKEL-22,160
80 GOSUB2000
90 GETA$:IFA$<>""THENB$=A$
100 IFB$="K"ANDPEEK(L+1)<>160THENPOKEL,
32:L=L+1:GOSUB600:POKEL,81
110 IFB$="J"ANDPEEK(L-1)<>160THENPOKEL,
32:L=L-1:GOSUB600:POKEL,81
120 IFB$="M"ANDPEEK(L+22)<>160THENPOKEL
,32:L=L+22:GOSUB600:POKEL,81
130 IFB$="I"ANDPEEK(L-22)<>160THENPOKEL
,32:L=L-22:GOSUB600:POKEL,81
140 IFL+1=AORL-1=AORL+22=AORL-22=ATHEN8
000
150 IFL=8152THEN60
160 POKESO,0
170 GOSUB2010
500 GOTO90
600 IFPEEK(L)=46THENPT=PT+1:POKESO,231
```

given at the end of this article. The joystick version runs approximately twice as slow as the keyboard, due to the frequent subroutines required for joystick commands.

Instructions are given at the beginning of the game, after an introductory title and short tune.

The program is constructed in a series of subroutines to allow modifications to be easily added. The routines are explained below.

The maze generation section is lines 1000-1999. The color of the background is controlled by variable CL. Lines 1210-1230 add cross connections at the lower difficulty levels, controlled by variable DL.

The enemy is moved by lines 2000-2999. The movement is executed in line 2010. Lines 2500-2510 subtract one point from the score for every dot the enemy eats.

Lines 5000-5999 control the introductory title and jingle. The prompt for instructions is given.

Instructions are in lines 7000-7999.

The game termination sequence is given in lines 8000-8999.

The major control loop is at lines 60-160. Here the enemy is controlled, and the keyboard input obtained and analyzed. A joystick branch to lines 9000-9999 is used here for joystick control.

The meaning of most of the variables should be obvious from their context.

Strategy in Gobble! will become obvious after a few games. Study of the enemy's movement will reveal a very useful fact about the direction it takes. You may find it is not always a good idea to clean out a level before moving to the next.

Only one "life" has been used here, although more could be added. Also only one enemy has been added. At higher levels, more can be introduced, although the game slows down considerably if a large number are controlled.

Good luck, and start Gobbling!

To save wear and tear on the fingers, a copy of this game can be obtained by sending $3 (to cover first class postage and duplicating), a mailer, and a blank cassette to: Tim Parker, 66 McKitrick Dr., Kanata, Ontario, K2L 1T7

```
610 RETURN
1000 A(0)=2:A(1)=-44:A(2)=-2:A(3)=44
1001 CL=CL+1
1002 IFCL=1THENPRINT""
1003 IFCL=2THENPRINT""
1004 IFCL=3THENPRINT"
"
1005 IFCL=4THENPRINT""
1006 IFCL=5THENPRINT""
1007 IFCL=6THENPRINT"
"
1008 IFCL=7THENPRINT"":CL=0
1010 WL=160:HL=46:SC=7690:A=SC
1020 PRINT""
1030 FORI=1TO21
1040 PRINTSPC(1)"                          "
1050 NEXTI
1060 POKEA,4
1070 POKESO,143:K=FNA(4)-1:X=K
1080 B=A+A(K):POKESO,O:IFPEEK(B)=WLTHEN
POKEB,K:POKEA+A(K)/2,HL:A=B:GOTO1070
1090 K=(K+1)*-(K<3):IFK<>XTHEN1080
1100 K=PEEK(A):POKEA,HL:IFK<4THENA=A-A(
K):GOTO1070
1200 POKE8152,32:POKE8130,32
1210 FORZ=1TO(10-DL):X=FNA(16):Y=FNA(18
)
1220 POKE7704+X+Y*22,46
1230 NEXT
1240 POKE8174,160
1250 PRINT"SCORE=";PT;"";SPC(12)"HI="
;HS
1300 POKE7788,140:POKE7832,133:POKE7876
,150:POKE7920,133:POKE7964,140
1310 IFDL<10THENPOKE8030,DL+176
1320 IFDL>9THENS1=INT(DL/10):POKE8030,S
1+176:POKE8052,DL+176-S1*10
1500 RETURN
2000 A=8152:POKEA,102:K=2
2010 B=A+A(K)/2:IFPEEK(B)<>160THENGOSUB
2500:POKEB,102:POKEA,32:A=B:K=(K+2)+4*(
K>1)
2030 K=(K-1)-4*(K=0)
2040 PRINT"SCORE=";PT;"¢ "
2050 RETURN
2500 IFPEEK(B)=46THENPT=PT-1:IFPT<0THEN
PT=0
2510 RETURN
5000 PRINT"";SPC(5)"BONUS SCOR
E"
5030 PRINT"  500 POINTS BONUS!"
5040 PT=PT+500
5050 PRINT"SCORE=";PT
5060 FORWL=1TO100:POKESO,INT(RND(1)*128
)+128
5070 FORT=1TO10:NEXT:NEXT
```

```
5080 POKES0,0
5090 FORT=1TO1000:NEXT
5160 RETURN
6000 PRINT"";SPC(5);"***********
"
6002 PRINTSPC(5)"*          *"
6004 PRINTSPC(5)"* GOBBLE! *"
6006 PRINTSPC(5)"*          *"
6008 PRINTSPC(5)"***********"
6009 GOTO6500
6010 PRINT"¢¢¢¢
INSTRUCTIONS?"
6060 GETA$:IFA$=""THEN6060
6070 IFA$="Y"THEN7000
6120 RETURN
6500 FORWL=1TO3
6505 FORSC=1TO9
6510 READX:POKES0,X
6520 FORT=1TO100:NEXT
6530 NEXT:RESTORE:NEXT
6535 POKES0,219
6536 FORQ=10TOOSTEP-1:POKEV,Q:FORT=1TO1
00:NEXT:NEXT
6540 POKEV,10:POKES0,0:GOTO6010
6560 DATA215,201,228,207,215,219,207,20
1,219
7000 PRINT"        GOBBLE"
7010 PRINT"   A RANDOM MAZE WILL"
7020 PRINT"BE DRAWN ON THE SCREEN";
7030 PRINT"CONSISTING OF DOTS."
7040 PRINT"
  ON THE SCREEN, YOU"
7050 PRINT"ARE SHOWN AS q, AND"
7060 PRINT"YOUR TASK IS TO EAT AS";
7070 PRINT"MANY DOTS AS YOU CAN."
7080 PRINT"YOU SCORE ONE POINT"
7090 PRINT"FOR EACH DOT EATEN."
7110 PRINT"  ALSO ON THE SCREEN"
7120 PRINT"IS YOUR NEMESIS, SHOWN";
7130 PRINT"AS &. THIS TRAVELS"
7140 PRINT"THROUGH THE MAZE, ALSO";
7150 PRINT"EATING DOTS. YOU LOSE"
7160 PRINT"ONE POINT FOR EACH DOT";
7170 PRINT"IT EATS."
7180 PRINT"  IF THE & GETS CLOSE"
7190 PRINT"TO YOU, IT WILL EAT"
7200 PRINT"YOU, AND YOU LOSE."
7210 PRINTSPC(12)"HIT A KEY";
7220 GETA$:IFA$=""THEN7220
7230 PRINT"  THERE IS AN ENTRANCE";
7240 PRINT"TO THE MAZE THAT SHUTS";
7250 PRINT"AFTER YOU ENTER. ONLY"
7260 PRINT"ONE EXIT EXISTS. TO"
7270 PRINT"LEAVE THE MAZE, AND GO";
7280 PRINT"TO THE NEXT LEVEL, YOU";
```

```
7290 PRINT"MOVE INTO THE EXIT."
7300 PRINT"AS THE LEVELS INCREASE";
7310 PRINT"SO DOES THE DIFFICULTY";
7320 PRINT"OF THE MAZE."
7330 PRINT"BONUS POINTS ARE GIVEN";
7340 PRINT"AT LEVELS 5 10 AND 15."
7345 PRINT"
";SPC(10);"I=UP"
7346 PRINTSPC(5)"J=LEFT   K=RIGHT"
7348 PRINTSPC(10)"M=DOWN"
7350 PRINT"
TRY FOR A HIGH SCORE."
7360 PRINT"IT'S NOT THAT EASY...."
7370 PRINT"  HIT A KET TO START";
7380 GETA$:IFA$=""THEN7380
7390 RETURN
8000 POKEL,102:POKEA,32:CL=0:POKESO,0
8010 PRINT"       GAME OVER    "
8015 IFPT>HSTHENHS=PT
8020 FORT=1TO1000:NEXT
8030 PRINT"   ANOTHER GAME?"
8040 GETA$:IFA$=""THEN8040
8050 IFA$<>"Y"THENEND
8090 PT=0:DL=0
8100 GOTO60
READY.
```

Assembly Language Programming on the VIC-20 Part II: Assemblers and Monitors

by Eric Giguere
Alberta, Canada

In last month's article I introduced you to three new concepts: binary numbers, machine language, and assembly language. This month we are going to look at an integral part of assembly language programming: assemblers and monitors. But before I get to either of these, which are very necessary for serious programming, I'm going to introduce you to another new numbering system. That's right, I said **another** one. So hold on to your hats and get ready to learn the **hexadecimal numbering system.**

Hexadecimal

If you remember last month's article, you'll recall how I said that binary numbers were simply numbers represented in base two, as opposed to decimal's (our system) base ten. Hexadecimal, then, is simply a base **sixteen** numbering system, with sixteen digits used to represent the decimal numbers 0 to 15, just as binary uses the two digits 0 and 1 to represent decimal 0 and 1. But hold on there: decimal has only ten digits—0 to 9. So how are you supposed to represent hexadecimal's sixteen digits with only ten? Easy—we don't. Instead, we borrow a few letters (A to F) from the alphabet and use them to represent the six digits after 9. This means that the symbol A equals 10 in decimal, and that B equals 11. Following is a list of the hexadecimal (also called **hex** for short) digits and their representative decimal numbers:

Hex	Decimal
0	0
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15

As you can see, the first ten digits (0-9) are the same, but then the letters A to F are used to symbolize the decimal numbers 11 to 15. What happens when you get to 16 in decimal though? It's just like in binary, how when you add one to 1, you move over one place to the left and place a 1 there, with a zero following. In hex this only happens when you add one to F (15) to make it 10 hex. Watch it, though, because 10 in hex means 16 in decimal, not **ten.** This is probably one of the most confusing parts of learning hex, so figure 1 explains it in more detail if you don't quite understand it. Also, to be able to distinguish between hexadecimal, binary and decimal numbers, the following symbols will be used.:

$ - hexadecimal	ex.: $2F
% - binary	ex: %10110011
nothing for decimal	ex.: 21

Note that if there aren't any symbols in front of the number, then it is considered to be decimal. (By the way, these are the industry standards for representing bases.) So whenever you see a number preceded by the "$" sign, consider it to be hex; if it's preceded by the "%" symbol, it means that the following number is in binary (usually recognizable by the long string of 1's and 0's); and if it isn't preceded by anything then consider it to be decimal.

Now that we've learned how the hexadecimal numbering system works, perhaps we should discuss why we had to learn it. I mean, it's no use to learn a new numbering system if we won't use it. So let's do that. Now then, of what use do you think hex is? Do you see any advantage as to representing a number like 32 as $20 hex? If you do, then you're one smart cookie. First of all, there is convenience. After all, the most a byte can hold, 255, can be shown as $FF. Also, the largest address the VIC or PET can access, 65535, can be represented as $FFFF (if you don't know why then refer to figure 2). Why use decimal if hex can cut down the digits used? Secondly, and more importantly, each digit in a hexadecimal number represents four bits of a byte. That is, each hex digit represents **half** of a byte. To see how this works, let's convert the number $OF to binary. This done, we see that it is shown as %00001111. Now let's do the same for the number $F0, and we come up with %11110000. Notice how $OF (15) takes up the rightmost half of a byte (which is eight bits) and the number $FO (240) takes up the leftmost half. If we combined the two numbers, would the whole byte be filled? Let's try it: $FO + $OF = $FF = 255 = %11111111. We were right! The whole byte got filled up. It's the same thing with any other hex number, even $00 (both halves of the byte are filled with zeros, so the whole thing comes out to equal zero). That is the real reason behind using the hexadecimal system. It's simply easier for the computer to translate the numbers than if they were in decimal. And though you may not find it very useful now, as you progress you will find it extremely handy.

Note: For those of you that would like a conversion table for all three bases (binary, hex and decimal) I'm making one available that also in-

cludes methods for converting from one base to another without using a table. To receive the table, send your name and address along with $1.50 to the address at the end of the article. If you also have some questions you'd like answered or suggestions and comments for the column, I'll try to answer them either in a personal reply or as part of the column.

Monitors

Well, we're finally finished with the topic of the hexadecimal numbering system, so it's time that we get along with the main thrust of this month's article. And that brings us to monitors. Just what exactly are monitors? As you should be able to guess, monitors are used to monitor the insides of your computer. Monitors are probably the most basic tools you can use to program in assembly language, apart from using POKE statements from BASIC. This is because a monitor only lets you enter numbers and to only see numbers. So really, all a monitor lets you do is to program in machine language, entering the hex numbers that represent your assembly language program. So a monitor really sound basic, and it is, although it can be quite useful at times.

Monitors have a basic set of commands that allow you to do several things. The most basic ones that you will ever find are M,S,L,G,R and X. These one-letter commands allow you to display and change memory (M); save memory to tape or disk (S); load a program (memory) from tape or disk (L); goto (execute) a machine/assembly language program in memory (G); display the registers (R); and exit monitor to BASIC (X). These commands are pretty well obvious and need no real definition (we'll learn about the R command next month), and are used by all the monitors I have seen. You can also get what is called an **extended monitor**. This is the basic monitor I have described here plus some enhancements, such as Hunt, which goes through memory checking for all occurrences of a certain byte, or Transfer, which transfers memory from one location to another. An example of such a monitor that has all these extended functions is the VIC-MON Machine Language cartridge sold by Commodore. It provides a lot of useful commands for the assembly language enthusiast at a reasonable price. A note for all you PET owners: all PETs except ones with Original ROMs in them have a built-in monitor with the basic commands. It is accessed by typing SYS 1024 from BASIC. Unfortunately for all you VIC owners out there, your VIC has no such toy built in. You have to buy one, either on tape or on cartridge. Any programs that I present in this column if monitor form will be usable by any monitor, so go ahead and buy the one that you like, because it shouldn't make any difference. All monitors are basically the same.

Assemblers

I mentioned that a monitor was a basic tool, really useful only for small programs or for programming in pure machine language. For the real serious programmer, though, only one thing is acceptable: an assembler.

An assembler is a program of sorts that lets you type in your assembly language program in **mnemonic** form instead of having to convert each instruction to numeral form and then place it into memory using the monitor. **Mnemonic,** you will recall, is a word that means "memory jogger", and refers to the three-letter instructions called **opcodes** (short for operation code) that tell the computer's CPU (central processing unit) what to do next. They are called mnemonics in that it is easy to infer their meaning just by looking at them. For example, the operation "load the accumulator with a specified byte" is shown as LDA in assembly language and as $A9 in machine language. If you had to work for a while with assembly language, wouldn't you want to use the assembler where you could type in relevent things such as LDA and STA instead of just numbers? I mean, after a while all those $FF's and $EF's will start to look alike, and you'll have a hard time trying to debug your program. so that's the first advantage that assemblers have over monitors.

The second advantage is that assemblers make it easy to document and update a program. You can put in labels and comments throughout your program to make it meaningful and easier to figure out when you take it out one day again and try to make some sense out of it. The use of labels is really helpful. Labels are sort of like line numbers in BASIC, except that they refer to memory addresses instead. You can define a label such as "LOOP1" to refer to the address 4123. You can also use labels as reference points inside the program itself. You see, an assembler usually has four **fields**, or areas in which to enter your program. The first field is called the label field, and is where you place the label that you have chosen to represent that part of the program. That label will then refer to the address of the mnemonic following label in the mnemonic field. Then whenever you want to go to that part of the program, you just put in a JMP (jump) instruction followed by the label of the area you want to go to. See figure 3 for an example.

Following the label field is the mnemonic field. This is the part in the line where you enter the opcode you want executed. All the codes in this field must have three letters, otherwise they are illegal, because all opcodes used by the computer are represented as three-letter codes. It's that simple. Now, following the mnemonic field is the operand field. In this field the assembler is given the instructions as to where to find the data that the opcode might need. For example, if you tell the computer to JSR (jump to subroutine), it has to know where the subroutine is. So what you actually code in is: JSR ROUTINE1. This will tell the computer to go to the subroutine specified by the label ROUTINE1. It's just like in a BASIC GOTO or GOSUB statement. If you don't tell the computer which line number to go to, it will return to you with a ?UNDEFINED LINE NUMBER error. So the operand field is definitely an important one.

Last and least is the comment field. This field is of no use to the assembler. All it is is a place for you to place your comments as to what this piece of code does or whatever. The important

thing to remember is that it is optional and does nothing for your program except take up memory. On the other hand, there are those who say it is indispensable because it makes for good documentation. Feel free to use it as you wish.

Whew! I think I'm getting a little too complicated. If you didn't quite understand what I was talking about, don't feel bad. It will all be clear to you as you progress. Suffice it to say that an assembler makes for easy assembly language programming, way easier than doing it by hand on paper and then converting the mnemonics to the right numbers. But for us VIC owners there is one problem: I've yet to see an assembler on the market for the VIC. Those of you with PETs have plenty of choices in this area (although they are all quite expensive), but us VICers have none. That is why I hope to be publishing my own assembler in BASIC soon. It will require a VIC with at least 8K, and perhaps even a disk drive, but for those of us who are serious in programming assembly language, it will be better than a monitor, that's for sure.

Figure 1: Notes on the Hexadecimal Numbering System—

In hexadecimal, you count just as you would in decimal, but instead of going 8, 9, 10, you go 8, 9, A, B, C, D, E, F, 10. There are six extra digits between the 9 and the 10. So each times you count to 10 in hexadecimal, you are actually counting to 16 in decimal. That is why hexadecimal is known as base sixteen, because your numbers are based on sixteen, not ten. The effect of this is that you have to count up to fifteen (F) each time before you "rollover" the digit. That is, when you get to the F digit, you add 1 to the next digit on the left and place a zero in place of the F. This happens in decimal, but only after you've counted up to nine, to get the decimal digits 10. To demonstrate this fact, let's count up to 20 in decimal: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20.

Each time the ones digit (the rightmost one) reaches nine, you add one to the digit on the left and clear the ones digit by placing a zero in it. This also works in hexadecimal, so let's count to 20 in that system: 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1a, 1b, 1c, 1d, 1e, 1f, 20.

You see? Each time you reach the letter F you clear it and add one to the digit on the left. But this time you had to add one thirty-two times before you got to 20. This means that 20 in hexadecimal actually represents 32 in decimal. Isn't that wonderful? (Actually, if you want a quick and sneaky way of converting hexadecimal to decimal, take the right digit and add it to the left digit which you multiply by 16. Ex.: 2A = 2*16 + A = 32 + 10 = 42.)

Figure 2: Numeral Positions and Powers of Sixteen

In hexadecimal, as in decimal and any other numbering system, the **position** of a digit is as significant as its **value**. In other words, the decimal number 10 is only different from 1 in that the position of the former is one place to the left more than the latter, and is thus worth ten times more its value. So basically, a digit represents a number equal to the digit's value multiplied by its base to the power of its position within the numeral. To put it another way, let us take the number 154 in decimal. To finds its value we would do the following:

$1 \times 10^2 = 1 \times 100 = 100$
$5 \times 10^1 = 5 \times 10 = 50$
$4 \times 10^0 = 4 \times 1 = 4$
Total 154

You see, what we did was take each number and multiply it by its base raised to a power that depended upon the digit's position. So for the digit 4 we multiplied it by 10^0 (the first position in a number is always referred to as the zero position) to get 4. The 5 was multiplied by 10^1 to get 50, and so on. Now then, this method doesn't just work with decimal. It works with any numbering system, including hex. So, to find the hex number $FFFF, we would do the following:

$F \times 16^3 = 15 \times 4096 = 61440$
$F \times 16^2 = 15 \times 256 = 3840$
$F \times 16^1 = 15 \times 16 = 240$
$F \times 16^0 = 15 \times 1 = 15$
Total 65535

And that's basically how you convert from hex to decimal.

Figure 3: Using Assembler Fields

An assembler has what is called four fields, or areas on a line that serve a different task. Something placed in one field will not work the same in another; they all have specific uses. The four fields go by the names Label Field, Mnemonic Field, Operand Field, and Comment Field. Each serves a specific purpose, as outlined below.

1) **Label Field:** This field is used to refer to a certain part of the program. It's sort of like line numbers in BASIC, but instead of line numbers you use letters or words. Its use is optional, but if you refer to a part of the program that you don't define, then your program could very well crash.

2) **Mnemonic Field:** This field is the most important. In here you place your instructions in mnemonic form. No options on this one.

3) **Operand Field:** This field is also very important. In in you give the computer the information as to where to

Next Month . . .

Well we sure learned a lot today (maybe too much!), and next month promises to be the same. The topic for next month's column should interest you. It's called "Registers and Addressing Modes" and is our first real leap into assembly language. In the meantime, if you get really interested about assembly language and can't wait to get my column each month, I suggest you get yourself a book or two on assembly language. It could prove very interesting.

For those of you who wish to contact me, my address is: Eric Giguere, Box 901, Peace River, Alberta, Canada TOH 2X0.

find the data it needs (as specified by the instruction in the mnemonic field). This data may be the label you gave to a specific part of the program, or it may simply be a number.

4) **Comment Field:** Totally optional, this field is reserved for you to enter comments to clarify the program. The assembler program completely ignores it.

Following are five lines of codes as could be entered on an assembler:

Label Field	Mnemonic Field	Operand Field	Comment Field
START	LDA	#0	;Start loop
	STA	STORAGE	;Store a value at the
	NOP		;location specified
	NOP		;by the label STORAGE
END	RTS		;return from subroutine

As you can see, the label fields were used only twice, as were the operand fields. The mnemonic NOP stands for a "No operation" and the computer just skips over it (sort of like a BASIC REM). Later on in the program we could call this subroutine by using its label as follows:

```
CALL        JSR          START          ;GOSUB to START
```

So you see, assembly language programming is made much easier by using assemblers. After you were finished typing in your program, the assembler would take it, convert the mnemonics and operands into numbers and then place them on disk or tape, or even in memory. Following that they can be loaded back into memory and executed.

USCD PASCAL for the Commodore 8096

by Neil Omvedt

If you read the hype about UCSD Pascal you might decide it is all good or all bad. What the pro's say is that it will force structured programming and that it is transportable from machine to machine. The major arguments against it are lack of input-output facilities and difficulty of learning.

My background in programming was heavily in PLI, a language that allows structured programming. So I decided to try out USCD Pascal to see if it could fulfill my needs. Before I got the product I did some reading and tried to write programs. It seemed like a cinch to go from PLI to Pascal, but later it turned out this was not quite so true.

The system is very complete. When you first load the program you get a prompt line which asks you what you want to do. The facilities available include the editor, the filer (for keeping track of data and program files), the compiler, the assembler, the program execution facilities and the linker. One thing you have to get used to is that it takes sometime for the system to switch back and forth between the various facilities. It is no worse than timesharing, but it is very predictable.

The editor is a very good screen editor and it allows you to do just about any editing you want. It takes a while to get used to, but it seemed fairly straight forward. My major problem was learning that you couldn't straight out edit on the screen, but had to choose one of the editor commands before you could do something. One shortcoming is that there is no help facility on the screen. You definitely have to refer to the manuals.

The documentation of the system is impressive. Almost 500 pages of manual. However most of the documentation relates to the UCSD p-system (the term for the total system). The documentation of Pascal is lacking, except for differences between UCSD Pascal and standard Pascal. You definitely need other books to learn the language. Certain areas are lacking. I could find no straight forward explanation of how to use both the screen and a printer for output in the same program. I still don't know if there is a straight forward way to do this. This to me is not a small item since in an interactive environment this becomes important.

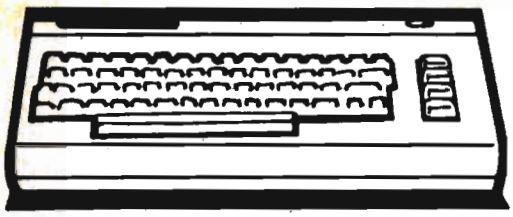
The last part of the documentation is about 10 pages from Commodore as to the use of the system. While it was enough to tell me how to setup my ASCII (read non-Commodore) printer for use with the system I have a feeling that more should be provided here.

With regards to Pascal itself the language syntax is very strict. All variables must be declared as to type (integer, real, string, etc.—you can even make your own types). I personally don't find this a problem, but beware if you are used to not declaring variables (as in BASIC). some of the syntax requirements I found difficult such as in the IF-THEN-ELSE clause. You do not use a terminator (a semicolon which marks the end of a statement) until after the else clause. Also a comparison such as "IF A=3 or C=5" is not valid. There must be parentheses around the subexpressions: "IF (A=3) or (C=5)" is the way it must be written. Suffice it to say that it is taking much more than the three days I took to go from FORTRAN to BASIC to make the change from PLI to Pascal.

I did run a small program to test execution time. Normally one would expect the Pascal to be considerably faster than BASIC because it is a compiled language. However, there is one hook in this. The UCSD Pascal system uses a pseudo-interpretr to translate the compiled code to use the 6502 CPU unit. This slows USCD Pascal down somewhat. My one test was a program to cube all the numbers from 1 to 10,000. The program took about one minute to run in Pascal and about one minute and twenty seconds to run in BASIC.

One other area of interest was the use of disk files. The first thing I found out was that the Pascal disk files are not compatible with BASIC. The Pascal defines three large files on the 8050 disk drives, and everything is internal to it. Thus a program datafile cannot easily be accessed from BASIC. There is probably some way around this, but it is not obviously easy.

All in all I am not ready to return my UCSD Pascal, but I don't find it quite as capable as I expected.



64

Peek & Poke

by George R. Gaukel
Tacoma, Washington

Sound Interface Device (SID)

The preliminary version of the advanced programer's manual for the 64 contains a lot of technical data on the SID. However, it gave few practical examples.

Due to popular demand, the listing for 'SOUNDER' follows. This is a BASIC program for exercising the SID and developing sound effects. The listing contains a few comments, as this was originally an exercise in menu management and interactive learning of the SID capabilities.

The program contains two displays, the main menu and the data menu. The main menu is a derivative of the proposed sound monitor menu in the preliminary manual. A second display was added after I determined that the original projection was not providing useful programing values. The 'LEFT-ARROW' will switch from one display to another. All main menu commands can be used while viewing the data menu.

Table 1 contains a list of the commands available. Many of the commands set a bit or nibble (4 bits). The combined byte (8 bits) can be determined by looking at the date display.

To get started with 'SOUNDER', enter the following:

SET 'V' TO 15 (VOL MAX)
SET 'G' TO 1 (GATE ON)
SET 'S' TO 15 (SUSTAIN)
SET '1' TO 1 (TRIANGLE)
SET 'F' TO 10,000 (FREQUENCY)

Now enter 'F7'. A tone will sound and will stay on until 'K'-Kill is entered. Set 'N'-Loop to 400 and enter 'F7'. Now the tone is killed after a delay. When 'F7' is entered, the SID will cycle through the attack, decay and sustain modes. The release mode is not entered until the gate is cleared/killed.

The keyboard input routine in the listing is a highly modified version of the CURSOR™ MAGAZINE input routine which they kindly placed the public domain.

I have found the 'SOUNDER' program useful in learning most of the capabilities of the SID and a time-saver in developing sound effects for other programs.

TABLE 1
SOUNDER COMMANDS

F1	Voice 1 Current Voice
F3	Voice 2 Current Voice
F5	Voice 3 Current Voice
V	Set Volume O-15
F	Set Current Voice Frequency O-65535
P	Set Current Voice Pulse Width O-65535
G	Set Current Gate ON/OFF
A	Set Current Attack 0-15
D	Set Current Decay 0-15
S	Set Current Sustain 0-15
R	Set Current Release 0-15
Y	Set Current Sync 0-15
M	Set Current Ring Mod 0-15
I	Set Current Filter ON/OFF
1	Select Triangle For Current Voice
2	Select Sawtooth For Current Voice
3	Select Pulse For Current Voice
4	Select Noise for Current Voice
O	Voice 3 ON/OFF (Combined Output)
N	Delay Loop for Kill (If = 0 Voice Stays On)
E	Resonance Factor 0-15
C	Active Filter Cutoff Factor 049151
L	LO Pass Filter ON/OFF
B	Band-Pass Filter ON/OFF
H	Hi-Pass Filter ON/OFF
X	External Voice Filter ON/OFF
F7	Sound Voices (Write Latches)
F7	Kill Voices (Clear Latches

```
100 REM 'SOUNDER'
110 PRINTCHR$(147):GOTO3000
120 :
130 HL=HXAND15:HH=(HXAND240)/16
140 HP$=MID$(HX$,HH+1,1)+MID$(HX$,HL+1,1)
150 RETURN
160 :
170 POKE RX,YY:POKE RY,0:POKE RP,0
180 SYS CU:RETURN
190 :
200 POKE RX,YY:POKE RY,XX:POKE RP,0
210 SYS CU:RETURN
220 :
```

```
230 GOSUB3570:GOSUB3500:GOSUB3410
240 RETURN
250 :
260 IF MU THEN230
270 YY=1:GOSUB170
280 PRINT" VOICE ([RVS-ON]F1[RVS-OFF] [RVS-ON]F3[RVS-OFF]
    [RVS-ON]F5[RVS-OFF])"VO"    [RVS-ON]V[RVS-OFF]OLUME:"RI
    GHT$(STR$(VE),2)
290 RETURN
300 :
310 PRINT"    [RVS-ON]F[RVS-OFF]REQ"BL$BL$"[RVS-ON]P
    [RVS-OFF]-WIDTH"
320 RETURN
330 :
340 IFMUTHEN230
350 I=VO:YY=4+VO-1:GOSUB170
360 PRINTRIGHT$(BL$+STR$(FR(I)),5);
370 HX=SI(I,FH):GOSUB130:PRINT" $"HP$;
380 HX=SI(I,FL):GOSUB130:PRINTHP$;
390 PRINTBL$RIGHT$(BL$+STR$(PW(I)),6);
400 HX=SI(I,PH):GOSUB130:PRINT" $"HP$;
410 HX=SI(I,PL):GOSUB130:PRINTHP$
420 RETURN
430 :
440 PRINT" [RVS-ON]G[RVS-OFF]T [RVS-ON]A[RVS-OFF]T [RVS-ON]
    D[RVS-OFF]E [RVS-ON]S[RVS-OFF]U [RVS-ON]R[RVS-OFF]L S
    [RVS-ON]Y[RVS-OFF] [RVS-ON]M[RVS-OFF]O F[RVS-ON]I
    [RVS-OFF] T[RVS-ON]1[RVS-OFF] S[RVS-ON]2[RVS-OFF] P
    [RVS-ON]3[RVS-OFF] N[RVS-ON]4[RVS-OFF]";
450 RETURN
460 :
470 IFMUTHEN230
480 I=VO:YY=9+VO-1:GOSUB170
490 PRINT " "RIGHT$(STR$(GA(I)),2);
500 PRINT " "RIGHT$(STR$(AT(I)),2);
510 PRINT " "RIGHT$(STR$(DE(I)),2);
520 PRINT " "RIGHT$(STR$(SU(I)),2);
530 PRINT " "RIGHT$(STR$(RL(I)),2);
540 PRINT " "RIGHT$(STR$(SY(I)),2);
550 PRINT " "RIGHT$(STR$(MO(I)),2);
560 PRINT " "RIGHT$(STR$(FI(I)),2);
570 PRINT " "RIGHT$(STR$(WT(I)),2);
580 PRINT " "RIGHT$(STR$(WS(I)),2);
590 PRINT " "RIGHT$(STR$(WP(I)),2);
600 PRINT " "RIGHT$(STR$(WN(I)),2)
610 RETURN
620 :
630 IFMUTHEN230
640 YY=13:GOSUB170
650 PRINT" VOICE-3 [RVS-ON]O[RVS-OFF]UT:"OT;
660 PRINT"    [RVS-ON]N[RVS-OFF]EXT LOOP: "RIGHT$(BL$+STR$(
    SD),5)
670 RETURN
680 :
690 IFMUTHEN230
700 YY=15:GOSUB170
```

```
710 PRINT" R[RVS-ON]E[RVS-OFF]SONANCE:"RIGHT$(STR$(RE),2)
720 RETURN
730 :
740 IFMUTHEN230
750 YY=17:GOSUB170
760 PRINT" A[RVS-ON]C[RVS-OFF]TIVE FILTER:"RIGHT$(BL$+STR$(
    CF),5)
770 RETURN
780 :
790 IFMUTHEN230
800 YY=19:GOSUB170
810 PRINT" FILTER SW: [RVS-ON]L[RVS-OFF]O:"LO " [RVS-ON]B
    [RVS-OFF]P:"BP " [RVS-ON]H[RVS-OFF]I:"HI " E[RVS-ON]X
    [RVS-OFF]:"XT
820 RETURN
830 :
840 IFMUTHEN230
850 YY=21:GOSUB170
860 PRINT " [RVS-ON]F7[RVS-OFF] SOUNDER   [RVS-ON]K
    [RVS-OFF]ILL  ";
870 RETURN
880 :
890 POKE53280,7+NOT MU
900 IN$=" ":ZT=TI:ZC=2:ZD$=CHR$(20)
910 GETZ$:IF Z$<>""THEN940
920 IF ZT<=TITHENPRINTMID$(" [\+]",ZC,1);"[CRSR-LEFT]";:ZC=
    3-ZC:ZT=TI+15
930 GOTO910
940 Z=ASC(Z$):ZL=LEN(IN$)
950 IFZKTHENIF(Z>132ANDZ<141)THEN1020
960 IF(ZAND127)<32THENPRINT" [CRSR-LEFT]";:GOTO1000
970 IFZXAND(ZAND127)>64AND(ZAND127)<91THENZ$=CHR$((Z+128)AN
    D255)
980 IFZL>9THENZ=141:GOTO1030
990 IN$=IN$+Z$:PRINTZ$;ZD$;Z$;
1000 IFZ=13THENIN$=MID$(IN$,2):GOTO1050
1010 IFZ=20ANDZL>1THENIN$=LEFT$(IN$,ZL-1):PRINT"[CRSR-LEFT]
    ";:GOTO910
1020 IFZKTHENIN$=Z$:GOTO1050
1030 IFZ=141THENZ$=CHR$(-20*(ZL>1)):FORZ=2TOZL:PRINTZ$;:NEX
    TZ:GOTO890
1040 GOTO910
1050 Z$=CHR$(-20*(ZL>1)):FORZ=2TOZL:PRINTZ$;:NEXTZ:RETURN
1060 :
1070 XX=20:YY=21+(2*ABS(MU)):GOSUB200
1080 ZK=1:GOSUB900
1090 IV=VAL(IN$):IFIN$=""THENIA=0:GOTO1120
1100 IFIN$=""THENIA=0:GOTO1120
1110 IA=ASC(IN$)
1120 ZK=0:PRINT" ";:RETURN
1130 :
1140 POKE53280,7+NOT MU
1150 PRINTPM$;
1160 PRINTRIGHT$(STR$(VT),2);
1170 PRINTZP$;
1180 GETIN$:IFIN$=""THEN1180
1190 IV=VAL(IN$):IA=ASC(IN$)
```

```
1200 IF IA=13 THEN 1260
1210 IF IN$="+"THENVT=VT+1
1220 IF IN$="-"THENVT=VT-1
1230 IF VT<0THENVT=0
1240 IF VT>LMTHENVT=LM
1250 VT=INT(VT):GOTO1150
1260 PRINT"          ";
1270 RETURN
1280 :
1290 :
1300 :
1310 REM**********************
1320 :
1330 PRINT"[HOME]";
1340 IF MU THEN GOSUB3340:GOTO1370
1350 GOSUB 3210
1360 :
1370 IFNOT MU THEN POKE53280,14:POKE53281,6:PRINTCHR$(5)
1380 IF MU THEN POKE53280,3:POKE53281,7:PRINTCHR$(149);
1390 GOSUB 1070
1400 :
1410 IF IA=95 THEN MU=NOT MU:PRINT"[CLR]":GOTO1330
1420 IF IA<128THEN1630
1430 :
1440 IF IA=133THENVO=1:GOSUB260
1450 IF IA=134THENVO=2:GOSUB260
1460 IF IA=135THENVO=3:GOSUB260
1470 IF IA<>136THEN1370
1480 :
1490 POKE Q0,QL
1500 POKE Q1,QH
1510 POKE FV,FS
1520 POKE PV,PW
1530 FOR I=1TO3:FORII=1TO7
1540 IFII=M5THEN1560
1550 POKE SM(I,II),SI(I,II)
1560 NEXT:NEXT
1570 FORI=1TO3:POKESM(I,M5),SI(I,M5)
1580 NEXT
1590 IF SD=0 THEN 1370
1600 FOR II= 1TO SD:NEXT
1610 GOTO 1820 : REM KILL
1620 :
1630 IF IV =0 THEN 1740
1640 IF (IV<1 OR IV>4)THEN 1370
1650 WT(VO)=0:WS(VO)=0
1660 WP(VO)=0:WN(VO)=0
1670 IF IV=1THENIV=16:WT(VO)=1
1680 IF IV=2THENIV=32:WS(VO)=1
1690 IF IV=3THENIV=64:WP(VO)=1
1700 IF IV=4THENIV=128:WN(VO)=1
1710 SI(VO,CN)=SI(VO,CN)AND150RIV
1720 GOSUB470:GOTO 1370
1730 :
1740 IF IA<65 OR IA>90 THEN 1370
1750 ON IA-64 GOTO 1970,2320,2840,2030,2460,2750,1850,2350,
     2520,1370
```

```
1760 ON IA-74 GOTO 1820,2210,1930,2970,2380,2660,1370,2150,
     2090,1370
1770 ON IA-84 GOTO 1370,2410,1370,2630,1890,1370
1780 GOTO1370
1790 :
1800 REM**********************
1810 :
1820 FORI=54272TO54296:POKEI,0:NEXT
1830 GOTO1370
1840 :
1850 SI(VO,CN)=SI(VO,CN)AND254
1860 IF GA(VO)=0 THENGA(VO)=1:SI(VO,CN)=SI(VO,CN) OR 1:GOSU
     B470:GOTO1370
1870 GA(VO)=0:GOSUB470:GOTO1370
1880 :
1890 SI(VO,CN)=SI(VO,CN)AND253
1900 IF SY(VO)=0 THENSY(VO)=1:SI(VO,CN)=SI(VO,CN) OR 2 :GOS
     UB470:GOTO1370
1910 SY(VO)=0:GOSUB470:GOTO1370
1920 :
1930 SI(VO,CN)=SI(VO,CN)AND251
1940 IF MO(VO)=0 THEN MO(VO)=1:SI(VO,CN)=SI(VO,CN) OR 4:GOS
     UB470:GOTO1370
1950 MO(VO)=0:GOSUB470:GOTO1370
1960 :
1970 VT=AT(VO):LM=15
1980 GOSUB 1140
1990 AT(VO)=VT
2000 SI(VO,DA)=SI(VO,DA)AND15OR(VT*16)
2010 GOSUB470:GOTO1370
2020 :
2030 VT=DE(VO):LM=15
2040 GOSUB 1140
2050 DE(VO)=VT
2060 SI(VO,DA)=SI(VO,DA)AND240ORVT
2070 GOSUB470:GOTO1370
2080 :
2090 VT=SU(VO):LM=15
2100 GOSUB 1140
2110 SU(VO)=VT
2120 SI(VO,RS)=SI(VO,RS)AND15OR(VT*16)
2130 GOSUB470:GOTO1370
2140 :
2150 VT=RL(VO):LM=15
2160 GOSUB 1140
2170 RL(VO)=VT
2180 SI(VO,RS)=SI(VO,RS)AND240ORVT
2190 GOSUB470:GOTO1370
2200 :
2210 IF LO=0 THEN LO=1 :GOTO2240
2220 LO=0
2230 :
2240 PW=PWAND15
2250 IF LO THEN PW=PW OR 16
2260 IF BP THEN PW=PW OR 32
2270 IF HI THEN PW=PW OR 64
2280 IF OT THEN PW=PW OR 128
```

```
2290 IFIN$="O"THEN:GOSUB630:GOTO1370
2300 GOSUB790:GOTO 1370
2310 :
2320 IF BP  THEN BP =0:GOTO2240
2330 BP=1:GOTO2240
2340 :
2350 IF HI  THEN HI =0:GOTO2240
2360 HI=1:GOTO2240
2370 :
2380 IF OT  THEN OT =0:GOTO2240
2390 OT=1:GOTO2240
2400 :
2410 VT=VE:LM=15
2420 GOSUB 1140
2430 VE=VT:PW=PWAND240ORVT
2440 GOSUB260:GOTO1370
2450 :
2460 VT=RE:LM=15
2470 GOSUB 1140
2480 RE=VT
2490 FS=FSAND15ORVT*16)
2500 GOSUB690:GOTO1370
2510 :
2520 IF FI(VO)THEN FI(VO)=0 :GOTO2550
2530 FI(VO)=1
2540 :
2550 FS=FS AND 240
2560 IF FI(1) THEN FS=FS OR 1
2570 IF FI(2) THEN FS=FS OR 2
2580 IF FI(3) THEN FS=FS OR 4
2590 IF XT THEN FS=FS OR 8
2600 IF IN$="I"THENGOSUB470:GOTO1370
2610 GOSUB790:GOTO1370
2620 :
2630 IF XT THEN XT=0:GOTO2550
2640 XT=1:GOTO2550
2650 :
2660 GOSUB890:IV=INT(VAL(IN$))
2670 IF IV>256^2-1THENIV=256^2-1
2680 IF IV<0 THEN IV=0
2690 VT=IV:PW(VO)=IV
2700 SI(VO,PH)=INT(VT/256)
2710 IF SI(VO,PH)<>0THENVT=INT(VT-SI(VO,PH)*256)
2720 SI(VO,PL)=VT
2730 GOSUB340:GOTO1370
2740 :
2750 GOSUB890:IV=INT(VAL(IN$))
2760 IF IV>256^2-1THENIV=256^2-1
2770 IF IV<0 THEN IV=0
2780 VT=IV:FR(VO)=IV
2790 SI(VO,FH)=INT(VT/256)
2800 IF SI(VO,FH)<>0THENVT=INT(VT-SI(VO,FH)*256)
2810 SI(VO,FL)=VT
2820 GOSUB340:GOTO1370
2830 :
2840 GOSUB 890:IV=INT(VAL(IN$))
2850 IF IV>49151THEN IV=49151
```

```
2860 IF IV<0 THEN IV=0
2870 CF=IV
2880 QH=INT(IV/256)
2890 IF QH<>0 THEN IV=INT(IV-QH*256)
2900 QL=IV
2910 QZ=QH AND 15:QZ=QZ*16
2920 QY=QL AND 240:QY=QY/16
2930 QL=QL AND 15
2940 QH=QZ OR QY
2950 GOSUB740:GOTO1370
2960 :
2970 GOSUB 890 :SD=ABS(INT(VAL(IN$)))
2980 GOSUB630:GOTO1370
2990 :
3000 Q0=54293:Q1=54294
3010 FV=54295:PV=54296
3020 V0=1:MU=0:POKE650,128
3030 PM$=" [RVS-ON]+[RVS-OFF] [RVS-ON]-[RVS-OFF]  "
3040 ZP$="[CRSR-LEFT][CRSR-LEFT][CRSR-LEFT][CRSR-LEFT]
    [CRSR-LEFT][CRSR-LEFT][CRSR-LEFT][CRSR-LEFT]"
3050 BL$=""+"      "
3060 HX$="0123456789ABCDEF"
3070 DIM SI(3,8),SM(3,8)
3080 M1=1:M2=2:M3=3:M4=4
3090 M5=5:M6=6:M7=7
3100 FL=1:FH=2:PL=3:PH=4
3110 CN=5:DA=6:RS=7
3120 CU=65520:RX=781:RY=782:RP=783
3130 :
3140 JJ=54271
3150 FORI=1TO3:FOR II=1TO7
3160 SM(I,II)=JJ+II
3170 NEXT:JJ=JJ+7:NEXT
3180 FORI=54272TO54296:POKEI,0:NEXT
3190 GOTO 1330
3200 :
3210 POKE53280,14:POKE53281,6:PRINTCHR$(5)
3220 GOSUB280
3230 PRINT:GOSUB310
3240 FOR I=1TO3:GOSUB360:NEXT
3250 PRINT:GOSUB440:PRINT
3260 FORI=1TO3:GOSUB490:NEXT
3270 PRINT:GOSUB650
3280 PRINT:GOSUB710
3290 PRINT:GOSUB760
3300 PRINT:GOSUB810
3310 PRINT:GOSUB860
3320 RETURN
3330 :
3340 POKE53280,3:POKE53281,7:PRINTCHR$(149);
3350 :
3360 PRINT"[HOME]";:GOSUB3420
3370 FORI=1TO3:GOSUB3510:NEXT
3380 GOSUB3570
3390 RETURN
3400 :
3410 I=V0
```

```
3420 XX=20:FOR I=1TO3
3430 YY=(I-1)*7:GOSUB200
3440 IFI=VOTHENPRINT"[RVS-ON]";
3450 PRINT"<<< VOICE"I
3460 NEXT
3470 YY=21:GOSUB200:PRINT"<<< COMMON"
3480 RETURN
3490 :
3500 I=VO
3510 XX=0:YY=(I-1)*7:GOSUB200
3520 FOR II=M1TOM7
3530 HX=SI(I,II):GOSUB130
3540 PRINTSM(I,II) RIGHT$(BL$+STR$(HX),3)" $"HP$
3550 NEXT:RETURN
3560 :
3570 XX=0:YY=21:GOSUB200
3580 PRINTQO;:HX=QL:GOSUB3640:PRINT
3590 PRINTQ1;:HX=QH:GOSUB3640:PRINT
3600 PRINTFV;:HX=FS:GOSUB3640:PRINT
3610 PRINTPV;:HX=PW:GOSUB3640
3620 RETURN
3630 :
3640 GOSUB130
3650 PRINTRIGHT$(BL$+STR$(HX),3)" $"HP$;
3660 RETURN
```

Peek & Poke #7
Mapping the Video Interface Chip (VIC)

The VIC is a very powerful video controller and we will see some excellent graphics on the 64 in the future. To use the VIC, we need to understand how the 64K of addressable memory in the 64 is seen by the VIC.

The VIC can directly decode only 16K of RAM/RPOM. This 16K block is selected by the 2 low bits of perphial register A of Complex Interface Adapter 2 (CIA2) at address 56576 ($DDOO).

BANK	BIT1	BIT0	ADDRESS RANGE
0	1	1	00000-16383
			$0000-$3FFF
1	1	0	16384-32767
			$4000-$7FFF
2	0	1	32768-49151
			$8000-$BFFF
3	0	0	49152-65535
			$C000-$FFFF

On power-up, both bits are set for decoding the low 16K of RAM or Bank 0.

The next control address is the Memory Control Register at address 53272 ($D018). The high nibble sets the visible memory within the selected bank in increments of 1024 ($400). On power-up the nibble value is 0001. This puts the visible memory at 1024 of the lower 16K Bank.

The lower nibble at the address (53272) contains the pointer for the character ROM in increments of 2048 ($0800). On power-up the nibble is 010x (two, the low bit is not used, so it always appears as a one). This value puts the character ROM at 4096 ($1000) in Bank 0. If the value is 011x, then the VIC is decoding the alternate character set at 6144 ($1800). The character ROM can only be decoded in Banks 0 & 2. Further, if the two high bits are not 01xx, then the character ROM will be ignored and RAM at this address will be decoded.

From the above, we can see that the address the VIC is latched to decode can be different from what the processor is currently decoding. While the processor is using the basic ROMs for program execution, the VIC can use the masked RAM for a video screen and the character ROM can be at the same address that the processor is using for BASIC RAM.

We now have progressed far enough to do some interesting things to the general memory lap. The program 'PETSYM' will remap the 64 to a PET memory map. A PET BASIC program that uses peeks & pokes to the screen can now be loaded and run on the 64. A PET program that peeks and pokes to other locations may crash and a PET program that uses 'SYS' calls will probably be a disaster.

```
100 REM 'PETSIM'
110 :
120 X=PEEK(56576) : X=XAND252OR1
130 :
140 REM FROM BANK 0 TO BANK 2
150 POKE 56576,X
160 :
170 REM ADJ VIDIO PAGE
180 POKE 53272,4
190 :
200 REM SET TOP OF BASIC MEMORY
210 POKE 643,255 : POKE 644,127
220 :
230 REM SET START OF BASIC MEMORY
240 POKE 641,1 : POKE 642,4
250 :
260 REM SET SCREEN MEMORY PAGE
270 POKE 648,128 : POKE 1024,0
280 :
290 REM SET MEMORY LIMIT
300 POKE 55,255 : POKE 56,127
310 :
320 REM SET START OF BASIC & POINTERS
330 POKE 43,1 : POKE 44,4 : NEW
340 :
350 :    CHANGES VIDIO MEM FROM $0400
360 :    TO $8000 (1024 TO 32768).
370 :
380 :    SETS CHAR ROM DECODE ADDRESS
390 :    FOR BANK 2 ($8000 TO $BFFF).
400 :
410 :    SETS START OF BASIC FROM $800
420 :    TO $400 (2048 TO 1024).
430 :
440 :    PROGRAM SELF-DESTRUCTS!!!
450 :
460 :     **********************
470 :     * SAVE BEFORE USING *
480 :     **********************
490 :
500 : RESTORE WILL NOT WORK PROPERLY
510 : AFTER RUNNING THIS PGM.
520 : POWER-OFF TO RESET THE C-64.
```

Commodore Character Set vs. ASCII

by Edwin Sund
Tacoma, Washington

What is ASCII? ASCII which is an acronym for American Standard Code for Information Interchange is a common code for representing a character set. How is ASCII represented? Usually it is a seven bit code (sometimes the eighth bit is used for reverse video). Broken down as follows:

BIT VALUES From	TO	
000 0000	000 1111	Device Control Characters
001 0000	001 1111	Device Control Characters
010 0000	010 1111	Punctuation and Special Characters
011 0000	011 1111	0 through 9 and Special Characters
100 0000	100 1111	@ and A through O
101 0000	101 1111	P through Z and more Special Characters
110 0000	110 1111	Apostrophy and Lower Case a through o
111 0000	111 1111	Lower case p through z and more Special Characters

As you can see there are eight groups of 16 characters for a total of 128 characters. As a standard for inter-computer communication this is just great because all computers know just what is being received and how to send information back.

With the advent of more sophisitication in data processing graphic characters were invented. But, alas, no standardization or convention has been universally adopted.

So what happened? Every computer manufacturer invented its own coding technique to represent its own graphics characters.

Well, Commodore, not to be outdone by any other manufacturer, has invented its own character set. Not just once, but twice and even in the same computer. Two sets you say? Yep, two sets! One set used to represent data entered from the keyboard and one set to represent data contained in screen memory.

Why was it necessary to invent a new code? There were now more than the original 128 characters which needed to be represented. Commodore characters are built using eight bits instead of seven giving it the capability of defining up to 256 unique characters. Because this was not enough, yet another 256 characters were defined by envoking an alternate set from a poke or in the case of the VIC and 64 by pressing the shift-commodore keys. Now, in the computer at any given nano-second one set of 256 characters is defined for the keyboard and one more set of 256 characters for the screen. Rather than list each group of characters the full list can be obtained in your reference book which you received with your computer. Let's now examine the difference between the keyboard characters and the screen memory characters. If you don't believe there

are two sets enter this!

Clear the screen.
Print CHR$(65)
Poke x,65
x = 32768 on a PET
x = 3680 on a VIC
x = 1024 on a 64

If you are in the normal character set you should have seen an "A" on the line after the print statement and a spade symbol in the top left corner.

Close study of the bit representations of the two sets will reveal the differences.

Let's look at the character "A". Keyboard representation is 0100 0001 = 41 HEX = 65 Decimal. Screen representation is 0000 0001 = 01 HEX = 01 Decimal. As you can see one of the bits is different between the two.

If we start counting from the left and count down the left most bit is bit number 7 and the right most bit is bit number 0. Bit 6 is on in the keyboard set and off in the screen set. Without belaboring this point the way screen characters are represented are by:

1. Get keyboard character
2. Drop bit 6.
3. Move bit 7 to bit 6.
4. If reverse is on, put A 1 in bit 7, otherwise put A O in bit 7.
5. Put the new character in the next position on the screen.

This is what happens every time a print statement is executed.

I hope the above explanation helps you understand how all this came about. Next month I will discuss how you can talk true ASCII from your computer to another computer or printer.

PET/CBM

REVIEW: A ROM for the PET "COMMANDER"

by Edwin Sund

The Commander is a toolkit like utility rom which performs many useful functions which are generally not found in micro systems with standard basic. The functions can be performed either within the program while it is running or as an immediate command.

The functions available are:

INSERT—Inserts lines into your program from another file.

APPEND—Adds a program to the end of your program.

DELETE—Deletes lines from a program.

COMMON—Defines common literals/arrays to avoid deleting them by an insert or delete command.

PRINTUSING—Allows a mask to be applied to a variable for use in formatted printing.

CONVERT—Converts PET ASCII into to standard ASCII.

FRAME—Provides information during long computations is. *** processing ***.

OVERLAY—Allows program overlays

ENHANCED GET—A series of 9 options for inputting alpha/numeric or numeric data.

RE-DIMENSION—Allows redimensioning of single dim. arrays

RETURN CLEAR—Clears all gosubs and for . . . next routines from the system stack and continues execution at any desired location in the program.

COMPUTED GOTO—Allows a goto based upon a computed variable.

WINDOW—Clears a "window" of lines-columns on the screen.

MAT PRINT#—Speed write to a disk file from an array.

MAT INPUT#—Speed read of a disk file into an array.

MAT INIT—Initializes array to " " or zero.

MAT ZER—Initializes array from specified starting point.

STRING—Inputs a string form disk (includes commas, etc.)

This rom requires a 4.0 pet and uses no additional memory to function. I have been using the COMMANDER for about a month now and feel very comfortable with it. However, at first, I was a bit confused. The problem was that I didn't use the demo disk provided which demonstrates all the uses of the commands. Additionally I was used to the TOOLKIT which uses key-words to perform the commands while COMMANDER uses sys statements to invoke the functions. SYS statements may sound confusing to use but once you have used the more commonly used functions the numbers are easy to remember. One benefit of using SYS statements over key-words like "append" is that if you use another utility rom the two roms won't fight over control of the system. There is no starting SYS statement to get COMMANDER to run. All in all I am very pleased with the added functionality and ease of developing COMMANDER based software. One final note, with a name like COMMANDER how could you go wrong!

TYPE-SHARE TYPESETTING
EDUCATIONAL SPECIALISTS
SILICON OFFICE SPECIALISTS

SOUTHERN CALIFORNIA
MAINTENANCE CENTER
FOR COMMODORE EQUIPMENT

ALL UNDER ONE ROOF

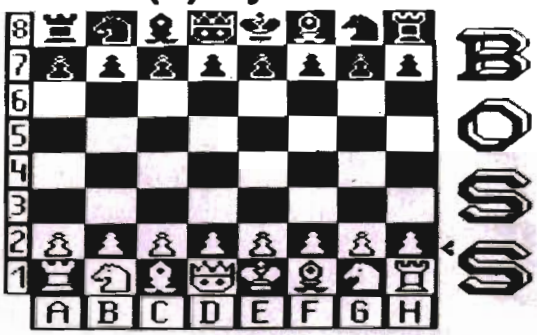
ONE STOP CENTER
for
commodore

VIC-VILLE ™ SOFTWARE

division of Data Equipment Supply Corporation

BOSS (c) by Kavan Software

Exclusive distributors of Kavan Software



The Definitive Chess Game for the VIC-20

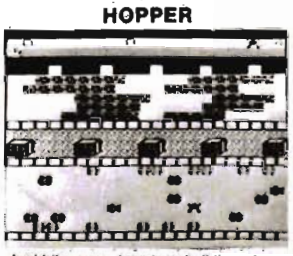
- ★ 10 Levels of Play
- ☆ Beats Sargon II
- ★ Two Clocks
- ☆ Wide range of opening moves
- ★ En passant, queening, castling
- ☆ Change screen and board colors
- ★ Cassette
- ☆ Requires 8K minimum expansion
- ★ 100% machine language

$39.95

BONZO (c) by Kavan

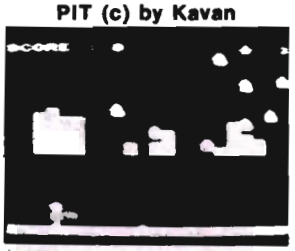
One of the most popular games in Europe. You control BONZO as he climbs the ladders and picks up the point blocks. Watch out for the alien guards. 100% machine language, cassette based. Joystick or keyboard, minimum 8k expansion. **$20.00**

HOPPER



Avoid the cars, dragsters, buildings, logs and other obstacles to bring the frog safely home. Machine language for fast and smooth arcade action. Joystick, standard VIC. **$20.00**

PIT (c) by Kavan



BONZO strikes again as he takes money bags out of the pit. Avoid the alien rain by standing under the shields. Every successfully removed bag of money reinforces your sheilds. 100% machine language, cassette based. Joystick or keyboard, standard VIC. **$18.00**

Commodore 64 YAHTZEE

Commodore 64 version of the famous dice game. 10 player capacity. Watch dice roll across the screen. Automatic tabulation of score and bonuses. Sprite graphics and sound. Cassette based. **$20.00**

Night Crawler **$25.00**
by Interesting Software
Shoot down centipedes, spiders, mushrooms and all kinds of bugs before they get you. Machine language arcade action on standard VIC with joystick.

The Black Castle **$20.00**
Adventure, travel the countryside, fight demons, buy goods, storm the castle. Requires 3k or more expansion.

A Maze Ing **$12.00**
Travel through the maze. Game of skill and tense action. Standard VIC.

Gobbler **$11.00**
Sounds easy? You have 25 seconds to get him and the time gets shorter at each higher level. Standard VIC.

Hang U **$12.00**
Traditional Hangman plays against the VIC's 250 word dictionary or another person. Standard VIC.

Coggle **$11.00**
Computerized version of Boggle. Standard VIC.

Gold Brick **$14.00**
Many levels of play, sound, and color.

3-D Labyrinth **$14.00**
Escape from the labyrinth. Shown in 3-D perspective view with randomly generated mazes. Standard VIC.

Air Strike **$11.00**
Fly the new super bomber V-20 on a mission. Standard VIC.

Attack on Silo III **$12.00**
You are the commander of Silo III. Defend your country. Standard VIC.

Baseball Strategy **$12.00**
The excitement of baseball as a video strategic game. Standard Vic.

Vic Poker **$14.00**
Play poker against the VIC. Hi-res graphics and sound. Standard VIC.

Frogger by (c) Kavan **$14.00**
Eat the flies and avoid the car. Standard VIC.

Space Phreeks **$25.00**
by Interesting Software
Pilot the spaceship "Infinity" and fight the "Space Phreeks". 15 different attack patterns, 33 levels. Machine language, arcade quality. Standard VIC, joystick.

Mailing List **$25.00**
Keep mailing list, print reports, labels. 8k expansion or 16k expansion required.

Astro-Miners **$17.00**
Hi-res graphics and sound space game. Requires 3k or 8k expansion.

Panzer Attack **$14.00**
Enemy tanks are attacking and you must destroy them. Hi-res graphics. Standard VIC.

Pedestrian Polo **$14.00**
Drive your car thru the streets. Based upon Death Race. Standard VIC.

Yahtzee **$12.00**
Solitaire version of this famous dice game. Standard VIC.

Commodore 64 Software **Available Now !!**
64 Monopoly from AP Software
64 Mailing List from VIC-VILLE™ Software
64 Finance from VIC-VILLE™ Software
64 Time Manager 2.0 from TOTL Software

Look for more 64 Software from VIC-VILLE™ & get on our mailing list for all 64 updates and users' group.

Complete descriptive catalog $2.00

Dealers Welcome - Authors Wanted !

Add $3.00 for shipping & handling

Network your CBM, VIC and COMMODORE 64 with the PET SWITCH and VIC SWITCH from DATATRONICS.

Distributors for Datatronic AB

(714) 778-5455 — DES — Data Equipment Supply Corp. 8315 Firestone Blvd., Downey, CA 90241 — (213) 923-9361

Given here, in zip code order, is a partial list of the Charter Dealers who will be carrying the COMMANDER. We will provide updates for this list in following issues as a service to provide our readers with a local source at which they will find information, hardware, or software for their Commodore Computers.

U.S.A.

Rhode Island

International Computer Services
165 Oyerville Ave.
Johnston, RI 02919
(401) 273-1001
Manager-Owner: Steve Lablanc

New Hampshire

Compucraft, Inc.
17 Dunbar St.
Keene, NH 03431
(603) 357-3901
Manager-Owner: Richard Bishop

Echo Consulting Services
P.O. Box 1199
Conway, NH 03818
(603) 447-5455
Manager-Owner: George Epotien

Maine

Maine Micro Systems, Inc.
55 Center St.
Auburn, ME 04210
(207) 786-0696
Manager: Nancy Lecompte

Vermont

Computeam
205 Dorse St.
S. Burlington, VT 05401
(802) 862-2802
Manager-Owner: Mark Robinson

New Jersey

Computer Workshop
1200 Haddenfield Rd.
Cherry Hill, NJ 07013
(609) 665-4404
Manager-Owner: Charles Kolbe

Computerability, Inc.
441 Route 23
Pomton Plains, NJ 07444
(201) 835-0688
Manager-Owner: Dennis Mull

Software City
147 N. Kinder Ramark Rd.
Montvale, NH 07645
(201) 391-0931
Manager-Owner: C.M. Hatfield

BB/The Computer Store
216 Scotch Rd.
Trenton, NH 08628
(609)883-2050
Manager-Owner: Barry Brown

New York

B.C. Communications, Inc.
World Wide Electronics Dist.
207 Depot Rd.
Huntington St., NY 11746
(516) 549-8833

Durmac Cash Registers
1628 Erie Blvd. E.
Syracuse, NY 13210
(315) 472-4531
Manager-Owner: William McCarthy

Personal Computers, Inc.
3251 Bailey Ave.
Buffalo, NY 14215
(716) 832-8800
Manager-Owner: Frank C. Smeirciak

Pennsylvania

One Stop Computer Shope
65 N. 5th St.
Lemoyne, PA 17043
(717) 761-6754
Manager-Owner: Joanne Wright

Micro Age Computer Store
1352 Tilghman St.
Allentown, PA 18102
(215) 434-4301
Manager-Owner: Ed Eichenwald

Maryland

Professional Micro Service
100 W. 22nd St.
Balto, MD 21218
(301) 366-0010
Manager-Owner: James A. Breen

Tri-State Computers
1504 S. Salisbury Blvd.
Salisbury, MD 21801
Manager-Owner: Tom Weiland

Virginia

Virginai Micro Systems
13646 Jeff Davis Highway
Woodbridge, VA 22191
(703) 491-6502
Manager-Owner: Shelli

West Virginia

Computer Associates, Inc.
113 Hale St.
Charleston, WV 25301
(304) 344-8801
Manager-Owner: Jeff Knapp

North Carolina

The Program Center
3400A W. Wendover Ave.
Greensboro, NC 27402
(919) 855-8667
Manager-Owner: Rupert Fenequito

Bob West Computers
54 West Main St.
Brevard, NC 28712
(704)883-2595
Manager-Owner: Sylvia West

Florida

COMPUTECH
1415 Timberlane Rd.
Tallahassee, FL 32312
(904) 893-1743
Manager-Owner: Dan Evans

Random Access Computers
296 Nelgin Parkway
Ft. Walton Beach, FL 32548
(904) 862-7763
Manager-Owner: Joanne Dodd

Commander Dealers

Florida Book Store
1614 West University Ave.
Gainesville, FL 32604
(904) 376-6066

Osceola Computer
1300 Dakota Ave.
St. Cloud, FL 32769
(305) 892-1501
Manager-Owner: Raymond Barrieau

Computer Specialties, Inc.
701 E. Lincoln Ave., P.O. Box 1718
Melbourne, FL 32901
(305) 725-6574
Manager-Owner: Otis P. Lutz

Focus Scientific
224 N. Federal Highway
Fort Lauderdale, FL 33301
(305) 462-1010
Manager-Owner: M. Rienhardt

The Software Connection
5460 N. State Rd. 7, Suite 108
Ft. Lauderdale, FL 33319

Alabama

Tricelin Corporation
Route 1, Box 128
Bankston, AL 35542
(205) 689-4999

Tennesee

Metro Computer Ctr.
P.O. Box 1406
Chattanooga, TN 37401
(615) 875-6676
Manager-Owner: Wayne F. Wilson

Mississippi

Sunrise Persons Supplies
P.O. Box 38341
Corinth, MS 38834
(601) 287-4721
Manager-Owner: Felex Gathings

Kentucky

All Business Computers
Suite C-2317 Versailles Rd.
Lexington, KY 40504
(606) 253-2545
Manager-Owner: Bud Wilson

Stowehuewge Computer, Inc.
2026-29th St.
Ashland, KY 41101
(606) 359-0545

Ohio

Office Mart, Inc.
11151 East Main St.
Lancaster, OH 43130
(614) 687-1707

The Computer Store of Toledo, Inc.
18 Hillwyck Dr.
Toledo, OH 43615
(419) 535-1541
Manager-Owner: Al and Jackie Miller

Computer Showcase
5855 Youngston-Warren Rd. SE
Niles, OH 44446
(216) 652-2571

Waltz Photo
438 Sixth St.
Canton, OH 44701
(216) 455-9421

Wards Computers, Inc.
868 Ohio Pike
Cincinnati, OH 45245
(513) 752-2882
Manager-Owner: Carl Ward

Indiana

Alen's Jewelry & Loan Co.
130 E. 10th St.
Anderson, IN 46016
(317) 642-7978
Manager: Jerry Rubenstein

AVC Corporation
2702 Applegate
Indianapolis, IN 46203
Manager-Owner: Brent Enderle

A Computer Store
2140 N. Mithoefor Rd.
Indianpolis, IN 46229
(317) 898-0331
Manager-Owner: Skip Robbins

Computer People
900 Highway 212
Michigan City, IN 46360
(219) 879-8557
Manager-Owner: Harry Hopkins

Computer Corner
6722 E. State Blvd.
Fort Wayne, IN 46815
(219) 749-8338
Manager-Owner: Tom Kutina

Custom Software
3197 South 3rd Place
Terre Haute, IN 47802
(812) 234-3242
Manager-Owner: Vicki McEntaffer

Michigan

Comm Data
320 Summit
Milford, MI 48042
(313) 685-0113

Roseville Computer
25929 Gratiot
Roseville, MI 48066
(313) 772-0760
Manager-Owner: Tom Potter

Haney's Stereo, Inc.
15270 Gratiot
Detroit, MI 48205
(515) 752-8845
Manager-Owner: Paul M. Paul

Computer Mart
915 S. Dort Hwy.
Flint, MI 48503
(313) 234-0161
Manager-Owner: Pat McCollem

Computers and More
2915 Dretom
Grand Rapids, MI 49508

Computer Tutor
502 E. Front
Traverse City, MI 49684
(616) 941-5320
Manager-Owner: Caroline Garrick

Iowa

Micro Computer Applications
111 E. Church St.
Marshalltown, IA 50158
(515) 752-8845
Manager-Owner: Harold Montover

Wisconsin
Majic Business Systems
3519 W. Wanda Ave.
Milw, WI 53221
(414) 282-8072
Manager-Owner: Dennis Woitekaitis

Computerland of Madison
6625 Odana Rd.
Madison, WI 53719
(608) 833-8900
Manager-Owner: James Sullivan

South Dakota
Computerland Rapid City
738 St. John St.
Rapid City, SD 57701
(605) 348-5384
Manager-Owner: John Mattson

Illinois
The Software Store, Inc.
1767 Glenview Rd.
Glenview, IL 60025
(312)724-7730

Digital World
711 Army Trail Rd.
Addison, IL 60101
(312) 628-9222
Manager-Owner: Sam Gunda

B-A Computer Sys.
2 N. Batavia Ave.
Batavia, IL 60510
(312) 879-2350
Manager-Owner: Robert Appel

Rozel Industries, Inc.
7360 N. Lincoln Ave.
Lincolnwood, IL 60646
(312) 675-8960
Manager-Owner: Fred Whitlock

Fischer Scientific
4901 W. Lemoyne Ave.
Chicago, IL 60651
(312) 378-7770
Manager-Owner: A.C. Heidrich

Kappel's Computer Store
125 E. Main
Belleville, IL 62220
(618) 277-2354
Manager-Owner: Tom Kappel

Data Plus, Inc.
1706 Broadway
Quincy, IL 62301
(217) 222-65602
Manager-Owner: James Moore

Missouri
Common Wealth Computers
5214 Blue Ridge Blvd.
Kansas City, MO 64133
(816) 356-6502
Manager-Owner: Dick York

Kansas
Computer Business Machines
Officenter 357 S. Lulu
Wichita, KS 67211
(316) 267-1150
Manager-Owner: Mrs. R. Santoscoy

Nebraska
Central Office Equipment
2020 Central Ave.
Kearney, NE 68847
(308) 234-2515
Manager-Owner: Byron Hanse

Texas
Computer Home
431 East Ave. C.
San Angelo, TX 76903
(915) 653-7488
Manager-Owner: Brent DeMoville

Tex-Tech
3115 W. Loop S., #26
Houston, TX 77027
(713) 965-9977
Manager-Owner: Phil Ray

Computerland of Amarillo
2300 Bell St.
Amarillo, TX 79106
(806) 353-7482
Manager-Owner: Mark Trowbridge

Whole Life Distributors
965 Washington, #6
El Paso, TX 80203
(303) 861-2825
Manager-Owner: Tom Tarbart

Idaho
Electronic Specialties, Inc.
8411 Fairview Ave.
Boise, ID 83704
(208) 376-5040
Manager-Owner: Terry Romero

Colorado
Zero Page, Inc.
2380 Naegele Rd.
Colorado Springs, CO 80904
(303) 633-0211
Manager-Owner: David C. Cooper

Utah
Mnemonics Memory Systems
(DBA Mnemonics Computer Store)
141 E. 200 South
Salt Lake City, UT 84111
(801) 266-7883
Manager: Rick Giolas

Arizona
Personal Computer Place
1840 W. Southern Ave.
Mesa, AZ 85202
(602) 833-8949
Manager-Owner: Roger Smith

Nevada
PCS Computer
3900 W. Charleston, Ste R
Las Vegas, NV 89102
(702) 870-4138
Manager-Owner: Mickey Cole

California
Opamp Tech Books
1033 N. Sycamore
Los Angeles, CA 90038
(213) 464-4322
Manager-Owner: Alicon

Data Equipment Supply Corp.
8315 Firestone Blvd.
Downey, CA 90241
(213) 923-9361
Manager: Robert Johnson

Computer Palce
23914 Crenshaw Blvd.
Torrance, CA 90505
(213) 325-4754
Manager-Owner: Wen T. Huang

Fyrst Byte
10053 Whittwood Dr.
Whittier, CA 90603
(213) 947-9411
Manager-Owner: Darrell Miller

HW Electronics
19511 Business Center Dr.
North Ridge, CA 91324
Manager-Owner: Ronda

Data Systems West
421 West Las Tunas Dr.
San Gabriel, CA 91776
(213) 289-3791
Owner: Frank J. Mogavero

Consumer Computers
8314 Parkway Dr.
La Mesa, CA 92041
(714) 465-8888
Manager: Steve Scott

Calco Digital Equpiment Inc.
1919 Aple St.
Oceanside, CA 92054
(714) 433-4119
Vice President: Ronald N. Paperno

Quality Computer Center
801 S. Victoria St., #104
Ventura, CA 93003
(805) 642-1979
Manager-Owner: David Stewart

Micropacific Computer Center
5148 N. Palm
Fresno, CA 93704
(209) 229-0101

J. Snell & Co., Inc.
657 Mission St.
San Francisco, CA 94105
(415) 421-5898
Manager-Owner: James Snell

PC Computers
10166 San Pablo Ave.
El Cerrito, CA 94503
(415) 527-6044
Manager-Owner: Gary Guttebo

The Computer Room
230 Mt. Herman Rd.
Scotts Valley, CA 95066
(408) 438-5001
Manager-Owner: Gary Guttebo

The Computer Center Stores
930 Town & Country Village
San Jose, CA 95128
(408) 246-5710

Inland Elestro Mart
8624 California Dr.
Riverside, CA 95204
(714) 687-3776
Manager-Owner: Jack

The Radio Place
2964 Freeport Bl.
Sacramento, CA 95818
(916) 441-7388
Manager-Owner: Gary Stilwell

Ray Morgan Co.
554 Rio Lindo Ave.
Chico, CA 95926
(916) 343-6065
Manager: Dave Wegner

Computer Place
1698 Market St.
Redding, CA 96001
(916) 221-1312
Manager-Owner: John Fredricks

Oregon

SW Computers
1125 N.E. 82nd
Portland, OR 97220
Manager-Owner: Jerry

Edu-Tech
1575 N.W. 9th
Corvallis, OR 97330
(503) 758-5577
Manager-Owner: L. Clark/W. Brown

Washington

Computer Town
1215 Center
Tacoma, WA 98409
(206) 272-2271

Computer Corner
1610 N. LaVenture
Mt. Vernon, WA 98273
(206) 428-1840
Owner: Kirk D. Shroyer

Conti Electronics Ltd
C½o Afcon
140-14th
Blaine, WA 98230
Manager-Owner: G.W. Harder

Alaska

BG Systems Co.
204 East International
Anchorage, AK 99502
(907) 276-2986
Manager-Owner: Robert DeLoach

Micro Age Computer Store
2440 Seward Highway
Anchorage, AK 99503
(907) 279-6688
Manager-Owner: Jay Wisthoff

CANADA

Ontario

Electronics 2001
5529 Yonger Street
Willowdale, Ontario M2N 5S3
Manager-Owner: Chris Bennett

House of Computers
368 Eglinton Ave. W.
Toronto, ON M5N 1A2
(416) 482-4336
Manager-Owner: Mark Herzog

The Computer Circuit Ltd.
733 Richmond Street
London, Ontario N68 3H2

Quebec

Systems Ornic Ltd
999 deBouragogue
Sinte Foy, Quebec G1W 4S6
Manager-Owner: Yvon Labbee

Caleq Inc.
331 Sir Walter Lourier Blvd.
St. Lambert, Quebec J4R 2L1
Manager-Owner: Marcel Bourcier

Alberta

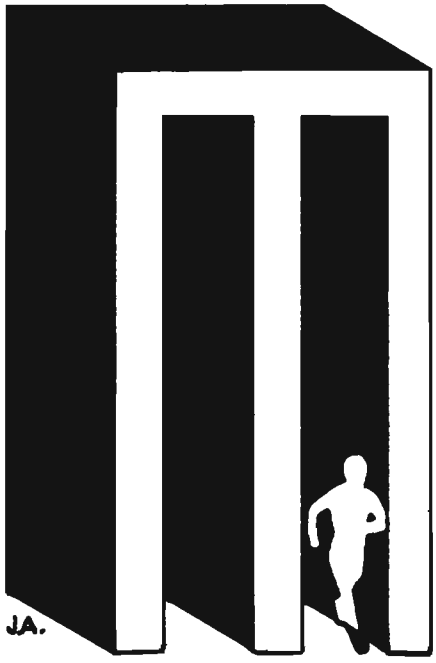
Kelley Software Dist. Ltd
P.O. Box 11932
Edmonton, Alberta T5J 3L1
Manager-Owner: Robert Owen

Game—CONTEST

Due to the newness of our magazine we are rerunning our game contest featured in our December Premier Issue. Next month we will have a new game contest.
The Editor

Escape MCP™

Introduced in late summer of this year by Comm*Data Computer House Inc., ESCAPE utilizes the full capacity of the VIC-20. Written in full machine code, requiring no additional memory and distributed on tape with a joystick option, ESCAPE MCP offers a simple challenge with no simple solution. In the program you have been de-atomized and teleported into the logic circuits of your computer. With you is the sinister MAIN CONTROL PROGRAM which wants to capture and destroy you. As you make your escape through the circuits, you discover that MCP chases after you, right through the walls! Try and try again as MCP taunts you with music and unbelieveable changing circuits. But you'll make it. After all you have speed and intelligence on your side! Besides your pursuer is Evil but still only a program. Make it through nine levels of logic and ESCAPE MCP. $14.95



The Game Contest is a continuing feature of Commander magazine aimed at providing entertainment for and promoting competition among our readers.

The Comm-Data company has graciously provided us with a great game and some super prizes for our Premier Contest.

Don't be the last one on your block to buy Escape MCP and beat the maze.

DEADLINE FOR ENTRIES: 1 MARCH 1983

Escape MCP may be purchased from COMM-DATA or anyone of its fine dealers.

P.O. Box 325
Milford, Michigan 48042
1-313-685-0113

Terms for the Premier Issue Contest

First prize will be awarded to the first person who successfully passes through all nine levels of Escape MCP. The winning entry must contain a photograph of the final screen of the game, an Escape MCP package front, proof of purchase slip and the correct name of the final tune played.

Entries must be mailed to COMMANDER , Escape Contest, P.O.Box 98827, Tacoma, Washington 98498. All entries must be mailed, as postmarks are required to determine the earliest winning entry. In the event of a tie duplicate prizes will be awarded. Employees of Comm*Data and their families may not participate. First prize will consist of Comm*Data VIC-20 Software, valued at $200.00. Second prize will consist of Comm*Data VIC-20 Software, valued at $100.00. Third prize will consist of Comm*Data VIC-20 Software, valued at $50.00.

The contest will run until a first prize is awarded. Comm*Data will notify Commander Magazine of the winner(s) and provide copy and photographs for a follow-up story.

Hints from the Commander

Zenith televisions and VIC's can be made compatible by typing in " POKE 3684, 133 " and pressing return. Zenith owners who wish to run Comm*Data's software must type in the poke **over** the " 3583 BYTES FREE" line that appears when the VIC is first turned on. The screen should look like this:

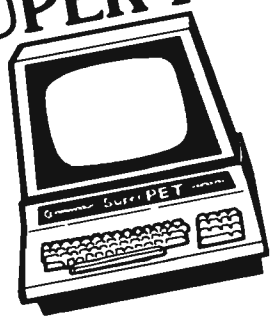
****CBM BASIC V2****
POKE 3684, 133
READY
LOAD

SUBSCRIBE TO *Commander*

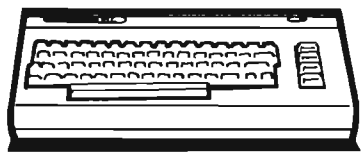
Now, and take advantage of our Charter Subscriber Discount of $4 OFF

THE MONTHLY JOURNAL FOR COMMODORE COMPUTER USERS

SUPER PET

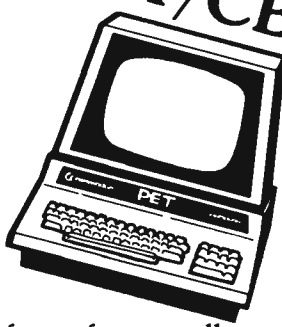


VIC - 20



64

PET/CBM



"COMMANDER will be dedicated to communicating the fun of, as well as the latest information about the ***COMMODORE COMPUTERS.****"*

EACH MONTH COMMANDER WILL HAVE:

- the latest information and news releases
- software for education, business and fun
- reviews on hardware and software
- program listings
- application (how-to) articles
- a contest and MUCH, MUCH MORE!!

DON'T MISS OUT

on the most informative magazine dedicated to the COMMODORE COMPUTERS

HAVE YOU GOT WHAT IT TAKES TO BE A *Commander?*

Advertising Index

LUNA SOFTWARE

LUNA SOFTWARE is quickly becoming one of the largest manufacturers of software for the Commodore 64™ and Vic 20™. The reason behind our growth is that we offer the consumer only the best in quality, service and price. We have now available for immediate delivery a diverse line of software — ranging from business programs to arcade style games. Call us today and shoot for the stars with LUNA.

DISKETTES & CASSETTES FOR THE COMMODORE 64™ AND VIC 20™.

FIREFIGHTER 64
STRATEGY GAME
Protect your homes and forest from the raging fire in this exciting strategy game.
Available in Cassettes for CM/64.

GAME GRAPHICS EDITOR
UTILITY PROGRAM
A must for all serious programmers. It lets you make multi-colored sprites at the touch of a button.
Available in Cassettes for Vic 20.

MAILING LIST
HOME/BUSINESS PROGRAM
Alphabetizing, search on any field, over 650 records per diskette, and much more make this the most powerful mailing list program available.
Available in Diskettes for CM/64 and Cassettes for Vic 20

PERSONAL FINANCE
PERSONAL PROGRAM
This versatile program can also be used by small businesses as an expense register and general ledger.
Available in Diskettes for CM/64 and Vic 20.

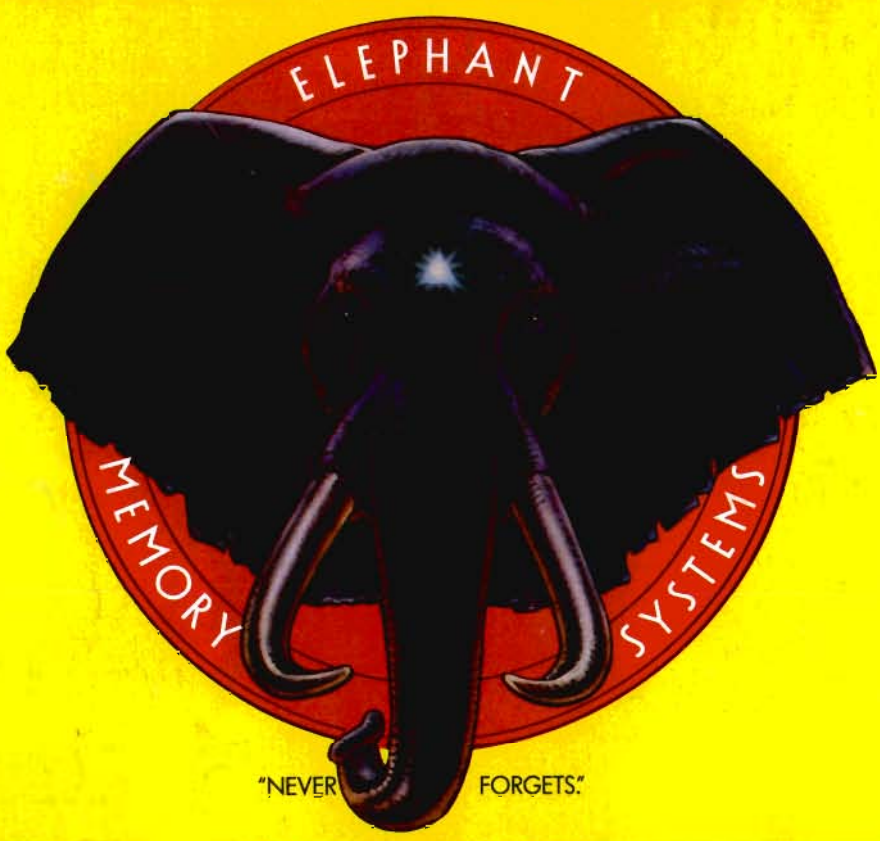
PEDESTRIAN
ARCADE STYLE GAME
Dart across busy boulevards and a perilous park as you avoid hazards and muggers in this fun and exciting game for all levels of players.
Available in Cassettes for CM/64.

...AND SO MANY MORE.
The above are only a few of the many programs that we have available.
To receive more information contact:

P.O. Box 26922
San Jose, CA 95159-6922
408-378-7793

LUNA SOFTWARE

REMEMBER:



MORE THAN JUST ANOTHER PRETTY FACE.

Says who? Says ANSI.

Specifically, subcommittee X3B8 of the American National Standards Institute (ANSI) says so. The fact is all Elephant™ floppies meet or exceed the specs required to meet or exceed all their standards.

But just who is "subcommittee X3B8" to issue such pronouncements?

They're a group of people representing a large, well-balanced cross section of disciplines—from academia, government agencies, and the computer industry. People from places like IBM, Hewlett-Packard, 3M, Lawrence Livermore Labs, The U.S. Department of Defense, Honeywell and The Association of Computer Programmers and Analysts. In short, it's a bunch of high-caliber nitpickers whose mission, it seems, in order to make better disks for consumers, is also to make life miserable for everyone in the disk-making business.

How? By gathering together periodically (often, one suspects, under the full moon) to concoct more and more rules to increase the quality of flexible disks. Their most recent rule book runs over 20 single-spaced pages—listing, and insisting upon—hundreds upon hundreds of standards a disk must meet in order to be blessed by ANSI. (And thereby be taken seriously by people who take disks seriously.)

In fact, if you'd like a copy of this formidable document, for free, just let us know and we'll send you one. Because once you know what it takes to make an Elephant for ANSI . . .

We think you'll want us to make some Elephants for you.

ELEPHANT.™ HEAVY DUTY DISKS.

For a free poster-size portrait of our powerful pachyderm, please write us.
Distributed Exclusively by Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021
Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.