THE TORPET

Second Class Mail Registration Pending

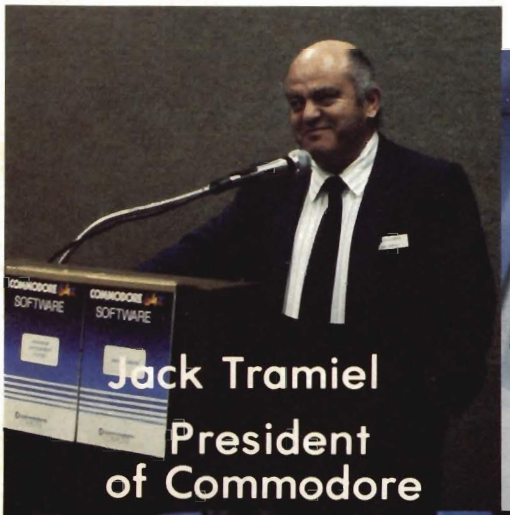
The INDEPENDENT Commodore Users' Magazine

No.17 February 1983 .

$2.00

New VIC & 64 Add-ons and Software

What Next From Commodore?



Jack Tramiel
President
of Commodore

Handheld Computer

The Millionth VIC

New Monitor for VIC & 64

Portable Computer

64 with Built-in Screen

VIC-20

The Friendly Computer

Commodore Vic 20 hardware and software
available at or through your nearest Eaton store.

EATON'S

CONTENTS

PAGE		
2	The New Generation of Commodore Computers	Ajay Jindal
4	Commodore at the Consumer Electronics Show	Chris Bennett
11	Whither the MAX?	Jindal and Bennett
14	Four VIC Games	Vince Sorenson
14	Two VIC Graphic Programming Utilities	Michael Kelinert
15	The VIC-20 Interfacing Bluebook	Ken A. Lee
16	Do It Yourself VIC Boards	Mike Liotta
18	VIC Hints	Vince Sorenson
19	Non Kernal Routines in the VIC-20	Thomas Henry
22	When Your Are Thinking of Buying a Computer	Mayland Harriman
23	Programming Contest	Michael Bonnycastle
27	Converting Programs from PET to 64	Garry Kiziak
32	PET - VIC - 64 Cross Reference Map	Mark Niggeman
34	Interfacing the Commodore 64 to a Video Monitor	John Ason
38	Some Serious Math	Jim Butterfield
39	The Hardware Hacker	Hank Mroczkowski
40	Superspeed Sort	David Williams
40	TORPET Information	Bruce Beach
41	The Lowly Cassete	Mayland Harriman
42	RTC Payroll Package	Stephen Smith
43	Meeting Reports	John Easton
44	A M.L. Presentation	John Easton
45	TPUG Info and Schedule	Chris Bennett
47	Traders Floor at TPUG Conference	David Williams.
48	Classified Ads	

The TORPET
published by
The Publisher
Horning's Mills, Ontario
Canada L0N 1J0
(519) 925-5376

Bruce M. Beach, Editor

Published monthly
Single subscription rate - $18.00 per year

Special bulk rates to clubs and dealers

THE NON COVER STORY
page 26

The New Generation of Commodore Computers

by Ajay Jindal
Scarborough, Ontario

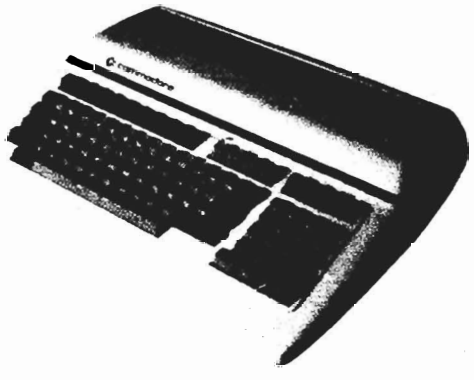
On Wednesday, November 17, 1982 I went to the Canadian Computer Show being held at the International Centre. At the show Commodore was showing the COMMODORE 64, VIC-20, PET, CBM, SuperPET and their new generation computers the P-500 and BX-700.

The information that I an going to pass on to you about the 'P', 'B', and 'BX' series computers (previously called the PET II and CBM II) is based on product specifications that were being handed out and what I picked up from the representative. This is only preliminary release information and specifications are subject to change.

The P-500

The Commodore 'P' 500 series (Advanced Personal Computer) looks much like an APPLE or ATARI 800 computer but has more keys (92 including separate numeric keypad and 10 function keys) and doesn't extend as far back. It is light grey in color and the keys are in two different shades of grey. There is no screen included so the video output is on a user-supplied monitor.

COMMODORE
'P' 500 SERIES
Advanced Personal Computer



It has a fullsized typewriter style keyboard with a separate numeric keypad. Above the keyboard and to the left are 10 function keys with which, by using the SHIFT key, you can get 20 functions. On the same level and to the right of the function keys are some more keys which are cursor controls, insert, delete, home and clear. Though I'm not sure, it is my impression that the above keys are not double function, (like on a PET where you press CRSR UP/DOWN alone to move the cursor down one row and with SHIFT to move it up a row). It seems that on the P-500 one key will be CRSR UP and one key will be CRSR DOWN, so you don't have to worry whether the SHIFT key is pressed or not.

The P-500 uses a new 6509 CPU that uses the same instruction set as a 6502 and has a 20 bit address bus. It is otherwise an 8 bit processor operating at 1.0 MHz. The language used is called CBM Extended BASIC 4.0 (ROM resident). It is essentially the same as the current BASIC 4.0 but some new commands have been added.

The P-500 will come with 128K of RAM expandable to 256K internally and 640K externally, tollaling a maximum of 896K of RAM. For the ROM there is an ambiguity. In one source it says there is a 40K ROM, and according to another source there is a 28K one. Both sources are from Commodore, so you can draw your own conclusions.

The interfaces found on the P-500 will be RS-232, IEEE-488, Dataset, Memory Expansion Slot and Connector, Dual Control Ports for Joystick/ Paddle/ Light Pen and an 8 bit user port.

Its graphics capabilities are much like the COMMODORE 64. It has character graphics, 320 x 200 pixels high res graphics (16 colours simultaneously), 8 sprites, smooth scroll and redefinable characters. The screen output is 40 x 25 colour characters and the character font is 8 x 8 pixels.

The sound on the P-500 is like the COMMODORE 64. It uses a 6581 SID chip with all the power of the COMMODORE 64. There might be an organ keyboard add-on with an extra synthesiser but there are no details at this time.

The P-500 can use all the disk drives a PET can including the 4040, 8050, 8250, and the D9060 and D9090 hard disks. The printers usable by it are 8023P, 4022P, 8300P or any other RS-232 printer. It can use any RS-232 or IEEE modem.

You can get an optional Z-80 processor for CP/M or an 8088 processor for CP/M-86. Other optional languages available are COMAL, UCSD PASCAL, LOGO and CBM Macro Assembler.

The P-500 can also take cartridges (24K ROM max.) but they are not compatible with either the VIC or COMMODORE 64.

The B-700

The Commodore 'B' 700 Series (Advanced Business Computer) looks like the P-500 but there is a screen and optional disk drives attached. The keyboard can be detached from the screen/disk unit.

COMMODORE 'B' 700 SERIES
Advanced Business Computer

The B-700 is pretty much the same as the P-500 but with changes making it more suitable for business.

I didn't get the handout on the B-700 but will give you information based on both the P-500 and BX-700 computers and general specifications given.

The first major differance is the video output. It comes with an 80 x 25 monochrome green screen with a character font of 9 x 14 pixels. There are no high resolution graphics but the graphic characters are still there. I imagine that there will be a high-res card available for the B-700.

There are optional integral twin drives (340K/680K) or you can use the regular external ones.

The last major difference is the absence of the joystick ports but there will probably be something set up so you can use a light pen.

The BX-700

The last New Generation computer from Commodore is called the Commodore 'BX' 700 Series (Advanced 16-bit Professional Computer). It looks the same as the B-700 computer with dual internal disk drives and green screen.

It comes with 256K of RAM expandable to 896K INTERNALLY.

COMMODORE 'BX' 700 SERIES
Advanced 16-Bit Professional Computer

The BX-700 houses dual processors. A 6509 with a 20 bit address bus and a 16 bit 8088. It operates at 2.0 MHz to 5.0 MHz. CP/M is available with the 8088.

Like the B-700 there is no joystick port and the video output is also the same as the B-700. Optional processors are Z-80 and 8087. You can get the languages compatible with them.

For the BX-700 there will be an experimental high-resolution card. The modems for the BX-700 must be RS-232 compatible.

Any other information on the BX-700 is either the same as the P-500 or was not available at the time of this writing.

According to the representative these computers are expected to be released around March 1983. Unfortuneately, I didn't get any prices and Commodore's office in Agincourt didn't have any prices either.

The pamphlet I received gives some general information on Commodore's complete microcomputer line and peripherals. Some stuff was given about the Max Machine (along with VIC-20, Commodore 64, PET, CBM, SuperPET, P-500, B-700 and BX-700).

Commodore at the Consumer

Commodore Booth at the CES January 6-9, 1983

The Winter Consumer Electronics Show was held in Las Vegas, January 6 to 9, 1983. All types of electronics were on display. My main interest was the computers on display, especially the new Commodore products. However, I did spend the first two days quickly looking at most of the other exhibits. These included audio/video, car audio, calculators, telephones, watches and many others too numerious to mention. This was the largest show of its type with over 75,000 people attending. However, since my main interest is Commodore Computers, I will restrict myself to that topic.

Since the show is a consumers' show, Commodore had only its consumer products on display. These are the VIC-20, Commodore 64 and all related products. No PETs or CBMs were on display. As of January 1st 1983, the Commodore 64 is considered a consumer product. This means that as soon as Commodore can produce enough, you will see them in the Sears, K-Mart type of store. This will not take place immediately, but will be done over the next 3 to 9 months.

Now for some good news. Commodore, in 1982, sold more computers than any other company. That includes: Texas Instruments, Atari, Radio Shack and Apple. With over One Million VICs and 75,000 Commodore-64s sold last year, COMMODORE IS NUMBER ONE in home computers. This was reported by Jack Tramiel who is the president and driving force behind Commodore International. Tramiel also predicts that by 1984, Commodore will be the number one manufacturer for all types of micros, including home, personal and business computers.

In 1981, Commodore had less than 1000 retail outlets. At the end of 1982, this was up to 12,000 outlets. By June of 1983, Commodore expects to have over 20,000 retail locations. Many of these locations will include toy, music and audio/video stores as well as the chain and computer stores presently carrying the VIC and C-64.

Zilog Z8000 CPU

Now for some specifics. Commodore interna-

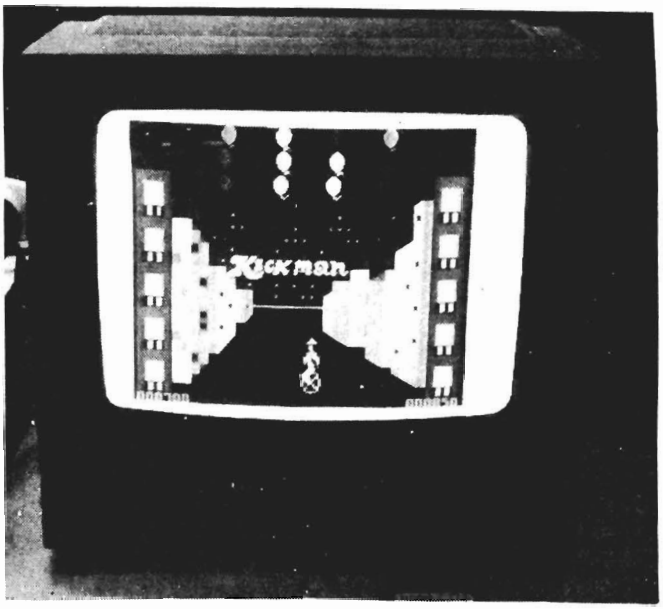
Electronics Show by Chris Bennett

Toronto, Ontario

Millionth VIC in Center Display

tional has signed an agreement with Exxon to second source the Zilog Z8000 CPU. This agreement covers both an exchange of technology and joint manufacturing. The Z8000 will be used in a new line of 16-bit computers that Commodore will be developing in the future.

The Exxon deal was necessary because Commodore will not depend on outside sources of Micro chips. The Z8000 is also a very powerful CPU, although it has not had the recognition of the 68000 or the 8086 CPUs. Zilog has a large amount of software available for the Z8000 including the CP/M and Unix operating systems. Now let us look at some of the new products that Commodore was showing. (PLEASE NOTE that all prices are quoted in U.S. dollars.)



The New Color Monitor

COLOUR MONITOR.

The CBM 1701 colour monitor is set to retail for $299.99 and is especially designed for use with the VIC-20 and C-64. The monitor accepts a standard 75 Ohm composite video signal or a

"Commodore" video signal with separate provisions for luminance and chrominance signal input as well as audio input. The monitor includes special circuitry which greatly enhances the picture resolution. First deliveries are expected by March 1, 1983.

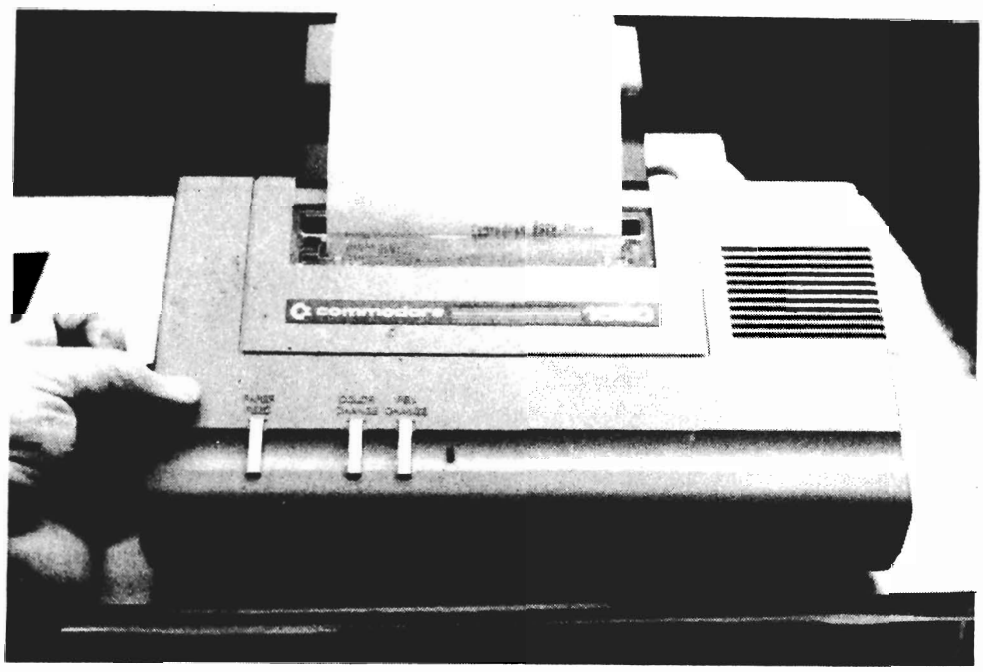
If you are interested in getting this monitor and have not bought a C-64 yet, then wait. The C-64's being sold up to now have a 5 pin DIN connection for the video output. The new C-64's have an eight (8) pin plug for video and only these versions will use all the new features of the colour monitor. The old C-64 will still work but may not be quite as good.

COLOUR PRINTER/PLOTTER.

The CBM 1520 is a printer/plotter priced at $199.95 for use on the VIC-20 and C-64. It prints on 4 1/2 inch roll paper and prints in four colours. High resolution graphs and charts can be drawn using the printer's ability to step 480 dots horizontally and up to 999 steps vertically. First units are scheduled for delivery during January 1983.

The 1520 can create text in any of four sizes, including 10, 20, 40 and 80 characters per line. I saw a sample output and even though the 80 characters on a line are very small, they are quite readable. The four colours available are BLUE, RED, GREEN and BLACK. Graphs are done very quickly with the pen moving horizontally across the paper and the roll paper moving foward and back to plot vertical dots. The 1520 can also print characters at abour 14 per second with accuracy to .2 millimeters.

Since this is a plotter, letters can also be printed sideways down the paper. The 1520 is an intelligent peripheral just like all COMMODORE devices. This means that the commands for drawing and plotting are INSIDE the printer. No special software is needed inside the computer, and all plotting can be done directly in BASIC.



4 color printer plotter

HANDHELD COMPUTER.

The HHC-4 is a handheld computer/calculator which can be used as a portable computer and full-function calculator or can be connected to a TV for full screen computing. The unit comes with 4K of RAM expandable to 16K with the addition of a plug-in memory expansion cartridge.

There is also a 20K ROM operating system which includes BASIC. The keyboard is a standard QWERTY type just like all other Commodore computers and also includes a separate numeric calculator keypad. Information is viewed on a built-in, 24 character, liquid crystal display or on a standard TV with the addition of a TV interface.

The HHC-4 can also be attached to a combination mini-printer and peripheral interface. The print speed is 24 characters per second. With the RS-232 interface, other printers and computers can communicate with this device. The HHC-4 will run about 300 hours on the 3 AA batteries. Although pricing had not been announced at the time, I heard about $100 for the computer and $100-150 for the printer/interface. Only time will tell if this is right.

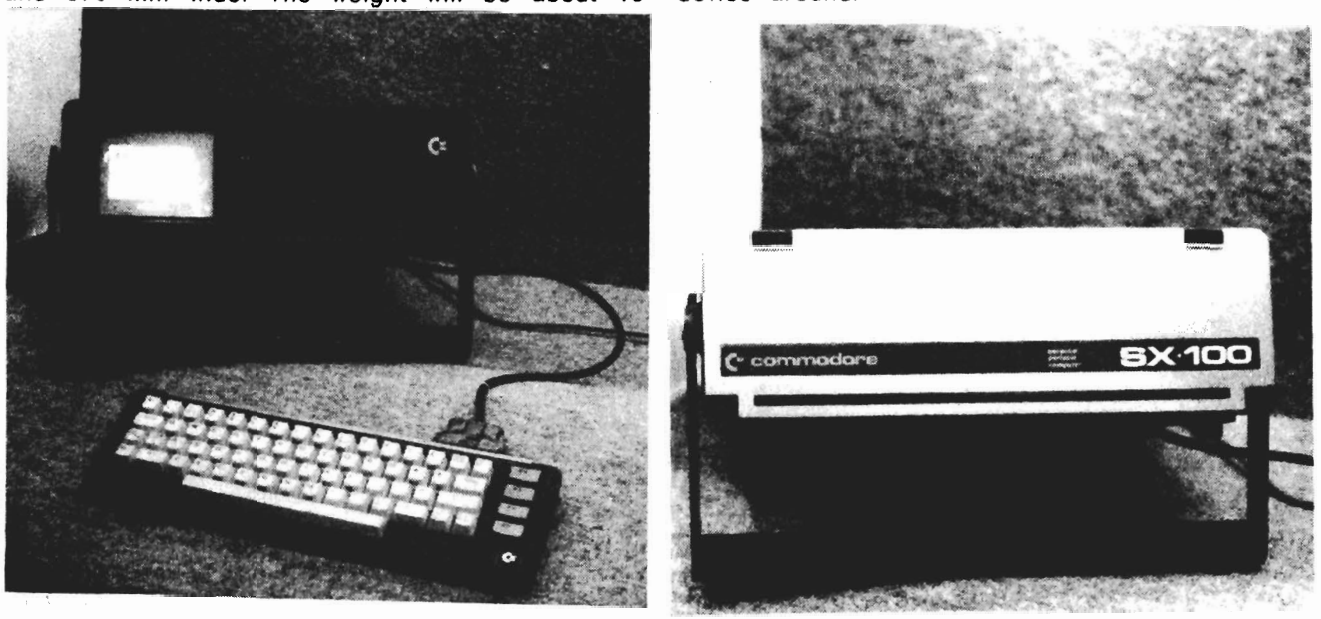
Handheld computer with printer.

PORTABLE COMPUTER.

The SX-100 is a computer-in-a-briefcase aimed to compete with the Osbourn portable computer. It is a Commodore-64 packaged with a 5 inch monitor and up to two 1541 disk drives. The black and white version with one disk drive will list for $995. The top-of-the-line will include a colour monitor and two disk drives and lists at $1595. Delivery is expected to be around April/83.

The SX-100 is 125 mm high, 370 mm deep and 370 mm wide. The weight will be about 18 Lbs. The device will also have the C-64 cartridge slot at the top, a cassette port, serial IEEE port and room to connect an external monitor. Although the 5 inch monitor is quite readable, I would not like to use it for extended periods of time.

Since the SX-100 is a packaged C-64, all the existing C-64 software will run on it, including games. The keyboard, like the VIC and C-64, is full-sized and has a very nice feel to it. It has a two foot cord connecting it to the main unit. It also serves as the cover when carrying the device around.



Portable SX-100 portable computer
both opened and closed.

New music keyboard for the 64.

MUSIC KEYBOARD.

This new product, due for release in May 1983 and not yet officially announced, is Commodore's entry into the music synthesis market. The system consists of three parts. First, a three octave keyboard which feels just like the keyboards you find on electric organs. Second, hardware which plugs into the C-64 cartridge slot. Finally, software to allow you to operate the keyboard.

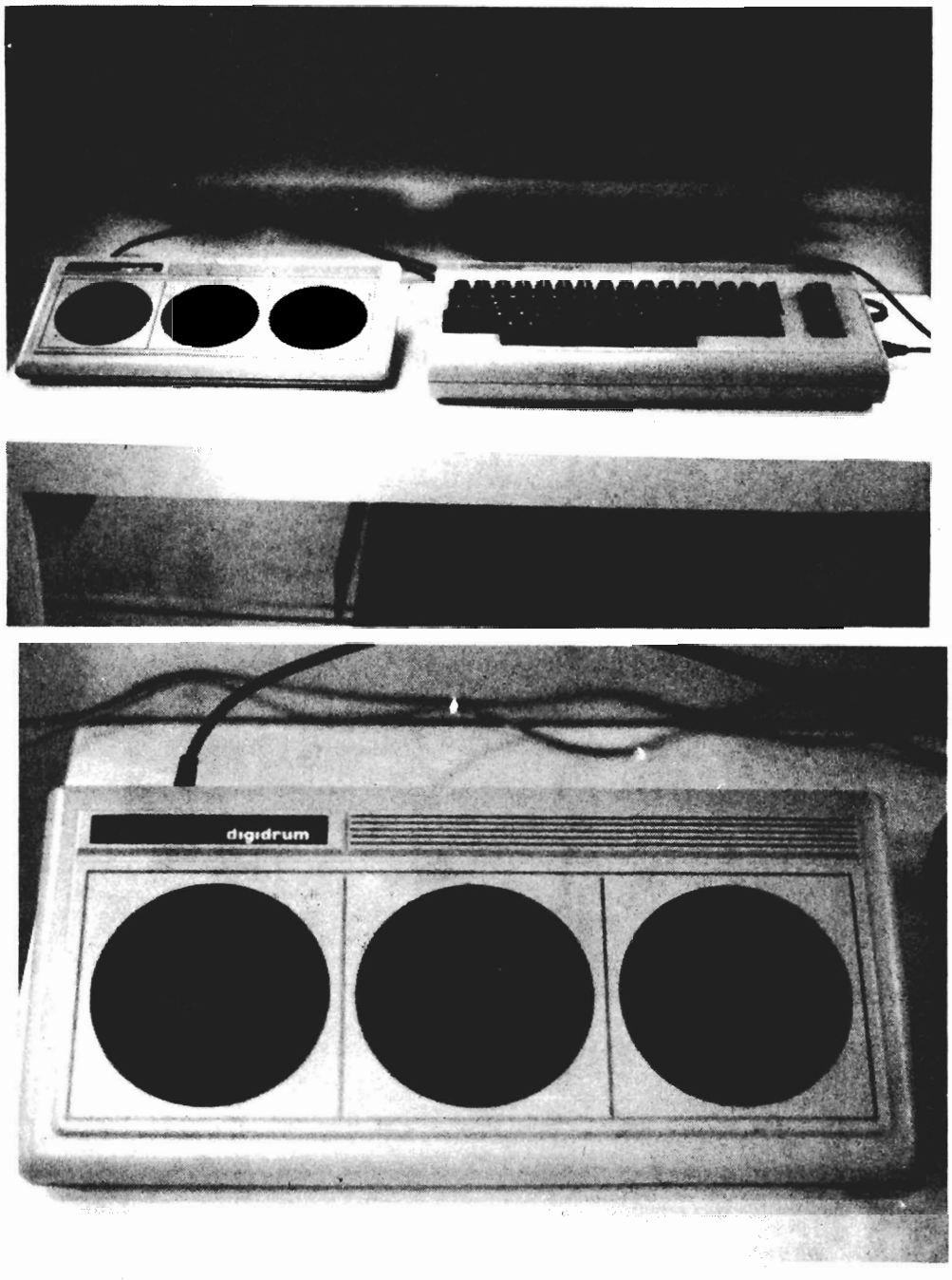
The complete package (keyboard, interface and software) will sell for under $100. I have even heard that Commodore may market the package at $69.95. Considering what is available on the market right now, even $100 will blow the competition away. The system is a twelve voice, music synthesizer, obtained by adding 3 more SID chips to the one already found in the C-64.

The VIC-20 version will have 4 SID chips in the interface since the music chip in the VIC is not as good as the C-64's SID chip.

The three octave keyboard is a full size organ style keyboard with a very nice feel to it. When I used it, it felt just like the keyboards on those $1000 organs. Of course, the keys are not pressure or velocity sensative like a piano. Hitting the keys faster or harder will make no difference to the sound. The software, developed by Commodore Canada's Paul Higginbottom, has a variety of features. There are 10 preset instruments built-in, each with a 7 octave range.

You can tranpose up 5, or down 7 semitones. The instrument can also be tuned up or down 7 increments of an eighth of a semitone each. Music can be saved in memory, on tape, or on disk. You can record one part of the music in memory and then play along with it.

David Berezowski from Commodore - Toronto much of the music systems development is being done in Canada.



Electronic drums for the 64.

Electronic Drums.

Another product which Commodore may release is a set of electronic drums. These drums have three rubber pads which, when hit, give the effect of some percussion instruments. I assume that this product uses the SID chip and thus the user may be able to set the type of instruments he or she wishes to use. More on this later as additional information becomes available.

SPEECH SYNTHESIZER.

In July of 1982, Commodore established a new speech technology division located in Dallas, Texas. Headed up by Dr. Richard H. Wiggins, the group was given the job of producing computer voice input and output for the Commodore line of home computers.

At the show was the first product out of this new division. It is a speech synthesizer which plugs into the cartridge port of the C-64. The cartridge contains special hardware chips to generate the sounds as well as memory (ROM) to contain 100 to 150 utterances in a female voice.

Also included with the package is an alphabet learning game for small children. The software also adds some commands to BASIC to allow you to include speech in your basic program. When you write your next space game, you can add some voice commands of your own.

This product is expected to be out later this year possibly in the $100 price range.

The modelling of sound is done by software not hardware. Therefore, many different types of sounds can be programmed. For example, I heard a male voice that was very natural sounding, a child's voice and even a sample of "Valley Girl speech. All of these sounds can be produced because the speech chips have D/A and filters built in. The software then models the sound, setting up pitch, timing and even inflection. This devise is only limited by the software modelling put into it. The quality of sound is excellent. It will make a good addition to the Commodore line.

RANDOM RAMBLINGS.

Commodore has sold 30,000 modems in the last three months of 1982.

Look for a $300 disk drive later this year possibly one of the new 3 inch drives. It is expected to connect to the cartridge port and run 5 to 10 times faster than the existing 1541.

Also possible, is a new dual drive to replace the 4040 disk drive. This drive would use the 1541 low profile drives and would be packaged in a much smaller box than the existing 4040.

Available mid January 1983 will be an IEEE single drive called the 2031SL. It will be a 1541 type drive with a full IEEE to connect to PETs and CBMs and will retail for the same price as the 1541.

The P-128 and B-128 have a list price of $795 with some machines being delivered in early February.

LOGO is in final testing as is CP/M for the C-64.

CP/M should be available in February with some public domain programs on disk.

The C-64 networking system is planned for delivery in March/83.

The 486-page Commodore 64 Programmer's Reference Guide has just been delivered to the major distribution centers and is now in the dealers hands. The cost is $19.95.

Other books being planned for release in early 1983 include: Making Music on the C-64, CP/M for the Commodore 64, and Introduction to BASIC, Part I and II. UMI announced price reductions of $10 each on game cartridges for the VIC-20. The new prices will be $39.95 and $29.95 down from $49.95 and $39.95.

Three components comprising the VIC modem, phone adapter and auto- answer system.

NEW SOFTWARE.

Commodore was showing more than 24 new pieces of software for the C-64 as well as many new cartridges for the VIC-20. Some of the new VIC software include: **COMMODORE ARTIST**, a lightpen drawing game; **SUPER SMASH**, a raquetball game; **CLOWNS**, a Bally/Midway arcade game; **SEA WOLF**, submarine against all enemy ships; **STAR POST**, a 3D space game; several children's games (**Home Babysitter, Tooth Invaders** and **Speed/Bingo Math**); **VICWRITER**, a wordprocessing program; **VICFILE**, a database package (requires disk plus 16K); **SIMPLICALC**, an electronic spreadsheet programs; **plus many more.**

The C-64 software includes many of the VIC games re-written for the C-64. These include **JUPITER LANDER, SEA WOLF, RADAR RAT RACE, AVENGER, OMEGA RACE, GORF** and others. Also shown were the **'EASY'** programs such as **EASYSCRIPT** and **EASYCALC.**

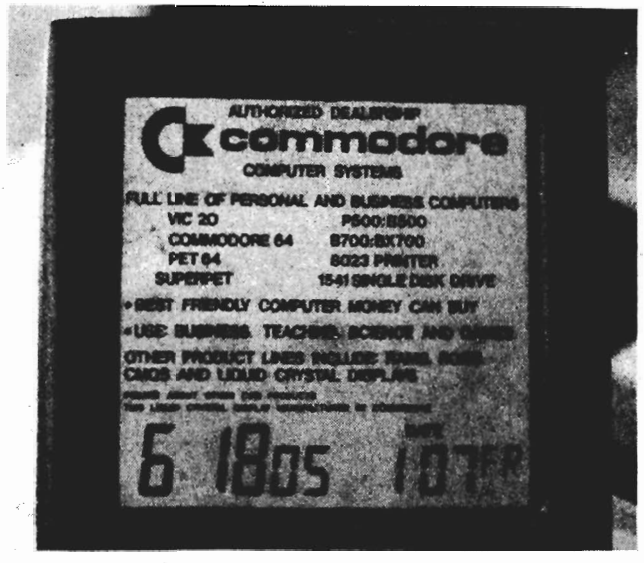
The **PET Emulator** should be in the stores soon so we can use all that free PET software in the TPUG library.

PAPERBACK BOOKS.

Commodore has introduced for the VIC-20 a series of software products which resemble paperback books in theme, packaging and content. The first five titles on display were: Know Your Personality, Know Your I.Q., Know Your Childs I.Q., Menu Planner and Quizmaster. The retail prices of these products will be from $20 to $40.

SUMMARY.

This was my first C.E.S. show, but not my last. I hope to drive out to Chicago for the summer C.E.S. show. From what I have seen and heard, the computer revolution is just about to start and the next few years are going to prove very interesting.



A large liquid crystal display. This may be the future.

Whither the MAX?

In case anyone out there is interested, here is some information about the Max.

It has 2K internal RAM memory plus whatever is in the cartridge. It does not have a fullsize typewriter keyboard but one that looks like a membrane (66 keys).

It is otherwise architecturally similar to the Commodore 64 but I don't know whether they will both use the same format of cartridges. It is described as a "Video Game with Computing Option".

It is due for release in early 1983 and will be priced around $100 US ($115 Canadian?).

-HOWEVER-

The MAX games machine is selling in Japan but there are no immediate plans to introduce it into the North American market at this time. Until Commodore can produce enough C-64's to satisfy demand, they will not bring out the MAX.

Smith-Corona introduces the first printer with real character at the unreal price of $1095.*

The Smith-Corona Daisy Wheel Printer

Until now, if you wanted to include a reasonably-priced printer as part of your computer or word processing system, you had to use a dot matrix printer. Daisy wheel printers were just too expensive.

Not anymore. Now Smith-Corona* offers a daisy wheel printer at such an incredibly low price, you can't afford *not* to include it. That means that even the smallest installation or business can now have letter quality printing capabilities at every work station.

The Smith-Corona printer operates with micro-processor-controlled daisy wheel technology, and is available with industry standard serial or parallel data interfaces.

Best of all, it produces results identical to those of our very finest office typewriters – printing with real character. So it can be used to create letters or documents that have to look perfect. As well as financial statements, inventory reports, direct mail campaigns – anything that requires quality printing.

And it's easy to use – just turn on the power, load the paper and away it goes. (It works equally beautifully with letterhead bond or fanfold paper.) There are drop-in ribbon cassettes and a choice of easy-to-change, snap-on daisy print wheels for a variety of fonts.

So why not get your hands on a real bargain: letter-perfect printing at an amazingly low price. Because, thanks to Smith-Corona, a printer with real character is no longer expensive.

Ask for it by name. Smith-Corona

*suggested retail price

DIVISION OF SCM® (CANADA) LIMITED

Please send me more information on the Smith-Corona daisy wheel printer.

Name_____

Title_____

Company Name_____

Business Address_____

City_____ Prov._____ Postal Code_____

Type of Business_____

Mail coupon or call:
Education Director,
Smith-Corona
29 Gervais Drive, Don Mills, Ontario M3C 1Z1
(416) 449-0164

T

Four VIC Games
VIC Software Reviews

by Vince Sorenson
Regina, Saskatchewan

CENTIPOD

Arcade game on tape for unexpanded VIC, needs joystick
from Commercial Data Systems,
730 Eastview Avenue,
Regina, Sask., Canada S4N 0A2
$29.95 (Canadian)

Centipod is a fast moving program very faithful to its namesake. Excellent graphics and sound add to the delight. Only complaint is the choice of colours on medium levels; you can hardly see what's happening. The author otherwise paid attention to small details, making this one better than 95% of the games I've seen. If you like arcade action and excitement, this one will captivate you.

Rating 4.5 out of 5.

SPIDERS OF MARS

Arcade-type Game on cartridge, joystick optional
from United Microware Industries (UMI),
3503 Temple Avenue, Suite C,
Pomona, CA 91768
$49.95 US ($60 Canadian)
(Writer's note - UMI have just reduced all their VIC software by $10)

Similar to Defender. Great use of the VIC's multi-colour hi-res mode. Action is fast, furious, and fun. Challenging but easy on starters. Only annoyance is that the joystick is difficult to use, sometimes too sensitive; sometimes not sensitive enough.

Rating 4.5 out of 5.

MINI-MOTHER

3 slot expansion board
from Quantum Data Inc,
3001 Red Hill Avenue,
Costa Mesa, CA 92626
$69.95 US ($80 Canadian)

It does the job its made for, but others can do it better, and for less. Does not have switching for cartridge selection, but is fused. It fits easily into the expansion port, and a styrofoam support, which protects it from damage it might otherwise easily get.

Rating 2.5 out of 5.

FROGEE

Arcade Game on tape for unexpanded VIC, needs joystick.
from Commercial Data Systems,
730 Eastview Avenue,
Regina, Sask., Canada S4N 0A2
$29.95 (Canadian)

Easy to learn, but still can challenge you at higher levels. Everything moves nicely, except the frog itself, which jumps when its on something that scrolls smoothly. Just like the arcade game, you cross a street and then a creek. Logs look like logs, the frog looks like a frog, and that alone puts it ahead of the Atari version. I'm still amazed that this company is local and hasn't yet put Saskatchewan on the map.

Rating 4 out of 5.

TWO VIC Graphic Programming Utilities

by Michael Kelinert
Nanuet, NY

One who has already attempted to program high-resolution graphics on the VIC, will have probably realized how inconvenient it is. The only way to do this on a VIC, without any graphics cartridges or software, is to define custom characters. This is very slow and tedious when plotting high-resolution circles, lines, individual points, etc.

There are now several hires graphic programming utilities available for the VIC in cartridge, ROM, and/or tape form. Two of the more popular ones in the US are the **Super Expander** cartridge from Commodore, and **GraphVics** from Abacus Software (an independant Commodore software dealer of Michigan). Both of these have some advantages and limitations.

The biggest difference to consider, and the one that probably causes most people to choose the Super Expander, is the fact that the Super Expander comes on cartridge and adds 3K of memory to the VIC. GraphVics, however, comes only on tape or disk and requires at least 3K memory expansion.

The main thing that probably makes GraphVics more appealing is its low list price of $25 ($30 for disk version), while the Super Expander lists for $59.95. (These are in U.S. funds).

They both have much in common. Both of them support commands for plotting individual points, drawing lines, drawing boxes, setting screen colors, and displaying text on the hires screen. However, the text display and box drawing commands in GraphVics are much better.

One significant difference between the two is that the Super Expander has special sound and music-writing commands while GraphVics has only hires graphic commands. The Super Expnder also has joystick, paddle, and light pen commands, and it allows you to define the function keys as commands or an entire line of BASIC. Two nice graphic commands on the Super Expander are the circle command, which allows plotting of circles of any height and width, and the paint command, which fills in an entire closed area.

GraphVics, however, has some commands that the Super Expander does not. One command places "tic marks" along the screen borders at specified intervals. This is nice for doing graphs. GraphVics also has the capacity of displaying large text in multicolor mode.

Another advantage with GraphVics is that you may plot points by giving coordinate values from 0 to 159. With the Super Expander, you must enter X,Y values ranging from 0-1023 which are then "scaled down" (as they put it) to values from 0 to 159. This is annoying because for any point you want to plot, you must first multiply the actual coordinates by 6.4 to figure out the corresponding values.

GraphVics has commands for saving and loading a hires screen to and from a device. It also defines two function keys for switching between hires and text screens.

Speed Comparison

In comparing the speeds of the two in plotting points and lines, I found that the Super Expander is 1.87 times faster at drawing lines and 1.54 times faster at drawing boxes. However, GraphVics is 2.77 times faster at plotting points. If you are going to draw circles, it would be much slower with GraphVics because it does not have a built-in command to do this.

Conclusion

GraphVics is a super buy at $25 but my only complaint is that it does not come on cartridge (it does come on disk for $30 U.S.). The Super Expander is a terrific cartridge but it is pretty expensive (it has built-in 3K expansion). If you have the money and you want the added 3K, sound, game controller, and function key commands, then you really should get the Super Expander. If you already have a 3K or 8K expander and you do not want to spend $59 for the Super Expander then your best bet is to get GraphVics.

Book Review

The VIC-20 Interfacing Bluebook

by Ken A. Lee Scarborough, Ontario

The VIC-20 Interfacing Bluebook sells for $14.95 + $2.00 U.S. postpaid. DON'T BUY IT!!!.

I've seen one and believe me its not worth it. The ads sound good but when you realize that the book is 46 pages of 5 x 7 inches with hand drawn circuits and diagrams and printed with a dot matrix format, you begin to see that there is no value for you money.

My problem with the book, aside from its format, is its contents. In my opinion, any book on interfacing any computer to its environment should contain certain information. This information includes:

1. Details of the input + output ports' physical structure.
2. Details of the electrical capabilities and tolerances to incorrect supplies and loads.
3. A discussion of timing, data and other signals.
4. Useful addresses and port control routines and programs.

Well don't expect anything like that. Of the 20 "Projects", all that is given is a few off hand remarks and freehand diagrams, along with sparsely commented programs. As a matter of fact, the memory expansion project does not even give a proper circuit diagram, and PCB templates are non existant. I wonder who would attempt this project with the muddy information presented?.

In summary, the ad for the book states that it costs less than 75 cents per project. I would agree that at 75 cents, maybe 1 or 2 projects would give the book a fair price of $1.50. On the other hand, your money would probably be better spent on a coffee and donut instead.

Do It Yourself VIC Boards

by Mike Liotta

The Vic-20 is quite a versatile machine for its price. However one of its limiting factors is the amount of memory available. Many programs reguire at least an 8K expander but even that doesn't leave a great deal of memory for data or text for such as a word processor. To solve this you can either go out and buy a 16K expander for $150-$170 or you can buy another 8K expander and expansion chassis to plug it all in. I have two inexpensive solutions for anyone handy with a soldering iron.

You can expand your 8K cartridge to 16K by simply soldering in more memory chips. Open up your cartridge and look at it from the component side. Refer to figure 1. Memory chips U1-U4 (in solid lines) are the ones on the boards now. U5-U8 (in dotted lines) will be the new chips. The board is clearly marked for their location. Table 1. shows the parts list. Although the Mitsubishi chips used by Commodore are not available the TMM 2016 chips will work fine. I got mine at Exceltronix for $9 each.

First thing you must do is remove the solder from the holes for the new chips and for capacitors C6-C9. I recommend that you use IC sockets. You will need the low profile sockets in order to get the board back in the case. Now install the sockets and capacitors C6-C9 and begin soldering. Use a fine tip on your iron and make sure that there are no solder bridges that will cause a short.

Now you can install the chips. Remember that these are CMOS chips, so don't touch any of the leads with your fingers. Either use an insertion tool or just be very careful. The orientation of these chips is the same as U1-U4. Location of the pin 1 is clearly marked on the board.

That's all there is to it. The jumpers beside switch SW 1 automatically allocates the new memory to location $4000-$5fff. There is no need for a new switch. If you purchase an 8K expander and install these chips you will save about $30 over the cost of a 16K cartridge. If you already have an 8K then the savings are much greater.

If this is not enough and you need more memory or you want to combine memory cartridges with utility cartridges such as Programmers Aid or Vicmon, you can build this simple 3 slot expansion chassis. The parts listed in Table 2 cost about $35 at Radio Shack. (Even less if you scrounge around electronics surplus stores).

Figure 2. shows the layout of the chassis as viewed from the top. Board A should be cut 5 cm. from the contact edge. Now cut out the holes on board B for sockets S1,S2,S3. Drill holes at locations D and E through both boards. You can now bolt both boards together. The bolt at location E is also used to hold the fuse holder. Now install the 3 card edge sockets and bolt them to the board.

Now comes the tedious part, wiring everything together. The easiest way is wire-wrapping using card edge sockets with wire-wrap posts.

The sockets are wired in a simple parrallel bus arrangement. Simply connect pin 1 of S3 to pin 1 of S2 and to pin 1 of S1. Then take the end of this wire and solder it to the solder tab for pin 1 on board A. Continue the same arrangement for all the other pins except for pin 21. This is the +5v line. The fuse is wired in series with this line between board A and the first socket. You have to be very careful when wiring the sockets because the pin numbers and letters seem to run backwards. If you buy board A from Radio Shack the first and last pin numbers and letters are labelled on each side. Refer to appendix J in the VIC Programmer's Reference Guide or appendix L in the VIC Mannual for detailed pin designations.

When all the wiring is done install bolts at location C. These act as feet for the chassis and must extend 1.5 cm. from the bottom of the board. Put a rubber tip on the end of these bolts to protect your desk. I used an eraser from a pencil. Now install a .5 amp. fuse in the fuse holder. This limits power to the chassis to 500 milliamps which is about all that the VIC's power supply can spare. This is enough for 16K of memory and 2 utility cartridges.

Before plugging the chassis in carefully check every single connection for shorts between any pins. Also check for continuity along each line. If everything checks out you can plug the chassis in. THe cartridges are plugged into the board with the top facing you.

You will need to pull the chassis out to use game cartridges with this arrangement. If you want you can design your chassis with switches to switch the game cartridges in and out. I have not done so.

If after all this you feel that you still need more then do what I just did. Go out and buy a commodore 64.

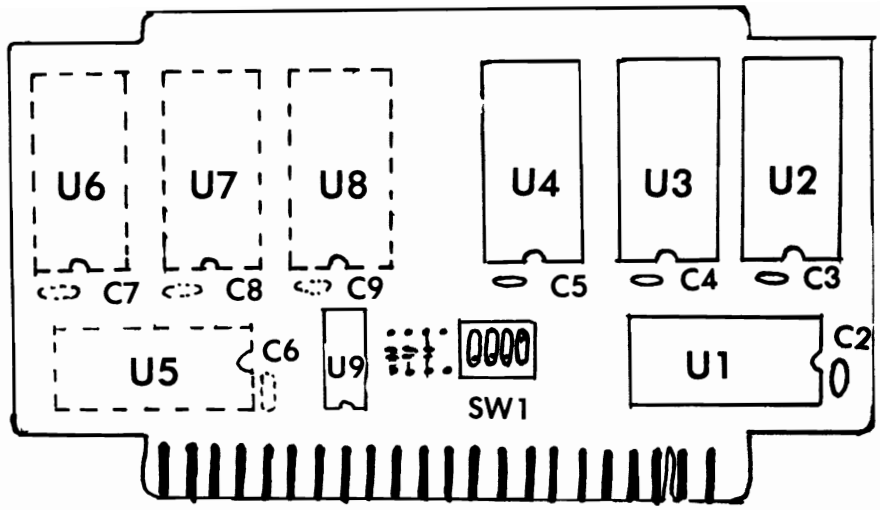


Figure 1

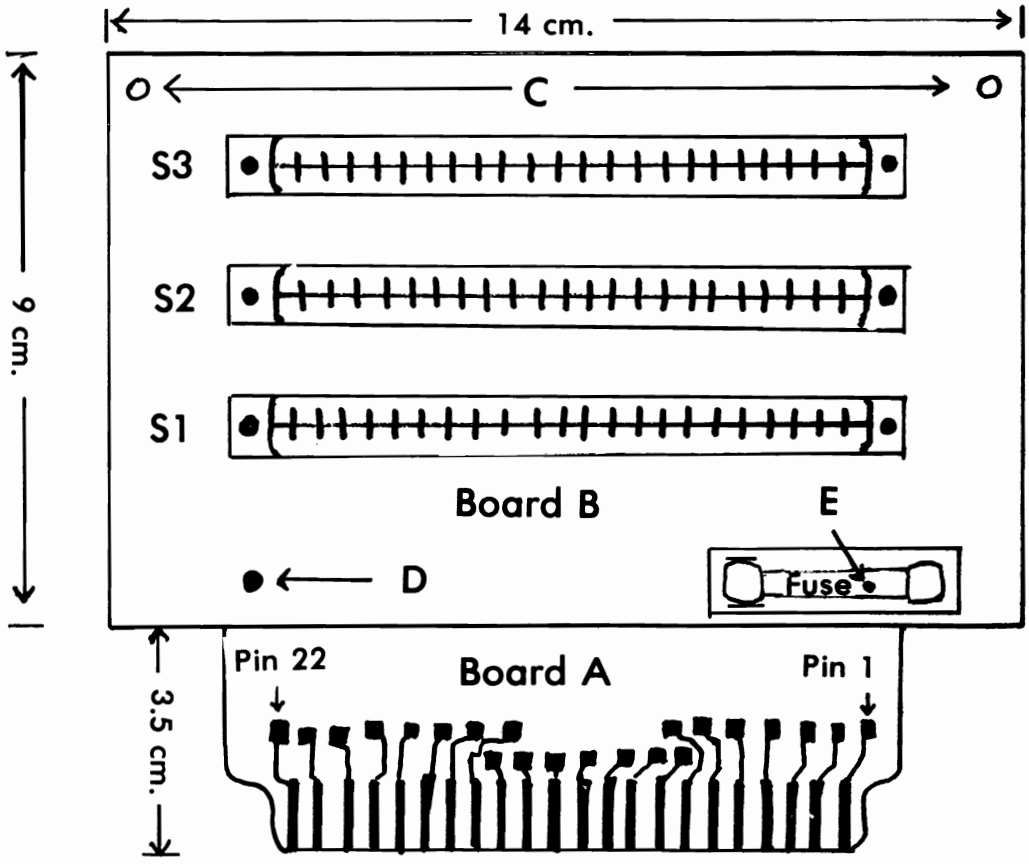


Figure 2

TABLE 1

Qty.	Part
4	TMM 2016 (2k X 8 Memory Chip)
4	IC Sockets
4	Capacitors (.1 uf.)

TABLE 2

Qty.	Part
3	Card Edge Sockets(44 pins R.S.#276-1550)
1	44 Contact Plug in Board (R.S.#276-156)
1	Perfboard (9 x 14 cm. R.S.#276-1394)
1	Fuse Holder (R.S.#270-739)
1	Fuse (.5 amp. R.S.#270-1271)

VIC Hints

by Vince Sorenson
Regina

After nearly two years of programming on the VIC, several ''tricks'' come to be used over, and over again. These ''tricks'' are used to save memory and speed-up program execution. Unfortunately, they can make your program listing unreadable, and difficult to change. Without further ado, I will explain these tricks, and their advantages/ disadvantages.

Trick #1
Location 37159.

Timer One is located here, and it controls how often the VIC goes to the IRQ routine. IRQ flashes the cursor, updates the time, checks the keyboard, and does several other things that take up most of the computer's time. If we can decrease the number of times the VIC uses the IRQ routine, we can speed up your programs in BASIC. How we do this is buy way of a poke into location 37159. Ordinarily, a 66 is in location 37159, but this can be changed. A higher number means the IRQ will be referenced less, and lower numbers means the IRQ will be referenced less.

You could think of this number as the last number of a FOR-NEXT loop used for waiting. Typing ''POKE 37159, 255(return)'' will speed yp your program's calculations. A side-effect of this, however, is that your cursor will blink and move slowly, the built-in clock will slow and become inaccurate, and keys tht are tapped too quickly may not be put into the keyboard buffer. So, location 37159 should be reset to normal when you use TI$.

Another side-effect is that pokes and peeks to the screen are done more slowly (pokes and peeks include print statements). While such things as entering data into memory for hires characters will be sped-up, the equivalent data will be put into memory for the screen more slowly. In the end, this trick is used most advantageously in adventure-style games, programs that calculate a lot, such as an alphabetizer, and some portions of action games.

Trick #2
Shifted-return
as a programmed cursor feature.

Any time that you type a shifted return in direct mode, your cursor returns to the next line, shuts off quote and cursor modes, but does not enter the line you were on. Return is exactly the same, except that it enters the line. Now, both return and shifted return can be coded or decoded as CHR$(X) or ASC(x$), where x$ is from a get statement, and X is 13 for return, and 141 for a shifted return. This is much like the other cursor controls, for instance, the delete and cursor down keys.

Delete is CHR$(20) and cursor down is CHR$(18). When you list these ''programmed cursor controls'', you see a reverse character, ''Q'' for cursor down,and ''T'' for delete. You should, therefore, be able to get a reverse character that represents return and shifted return. You can. The reverse ''M'' represents return, and the reverse shifted ''M'' represents shifted return. These can be but into a program line by typing in that line, pressing return, pressing CTRL-RVSON, positioning the cursor over the spot where you want your programmed cursor control, and typing the apprropriate key (''M'' for return, shifted ''M'' for shifted return). You should have left a space for this.

The effect of these controls is the same as id you had directly typed them in, except for certain differences. A programmed return will end your line at that spot as though you had never typed the rest of the line. However, the shifted-return will work exactly as the direct shifted-return, both in the display and listing. The line: 10 PRINT''HELLO,(RVS, shifted)M(RVSOFF) THERE'' -will be listed as:

10 PRINT''HELLO,
THERE''

and when your program is run, your display will have ''HELLO'' and ''THERE'' on separate lines. This trick is very handy for getting rid of waste spaces, and extra PRINT statements that take up memory. Unfortunately, to edit a line with a shifted return in it, you have to retype that entire line. Therefore, it is best if you put the programmmed shifted return in after in after getting the bugs out of your program.

Trick #3
Repeating keys

All keys will repeat if you type POKE650,128.

Trick #4
Easy Screen Move

An easy way to move the screen is to type POKE648,X and then RUN STOP/RESTORE. Your screen is now located at X*256 to X*256+505. This trick can be used to move the screen to 6656, leaving 7168 to 8191 for your character set (where the first 128 are user developed and the last 128 are the first 128 of the regular character set, instead of 64 user developed, 128 regular, and 64 waste characters).

You'll need to POKE56,26:POKE648,26:CLR and press RUN STOP/RESTORE afterward. One more thing you'LL need to know is where to find the colour memory. It is now at 37888+256*(PEEK(648)AND 2). See also Jim Butterfield's article in Home and Educational Computing (Compute!), for a way to move the screen under programm control.

NOTES

Other ways to save memory include using integer variables (but only when that variable is repeated in the program less than 7 times), and shorter variable names.

One last note: Trick #2 will work for any Commodore computer. Other tricks will work on various Commodore machines, but the corresponding locations must be found.

Good luck in your programming. I hope these ''tricks'' will help.

Non Kernal Routines in the VIC-20

by Thomas Henry

Transonic Laboratories Makato, Maine U·S·A·

The VIC-20 PROGRAMMER'S REFERENCE GUIDE, (Indianapolis, Indiana: Howard W. Sams and Company, 1982), covers the so-called Kernal routines in some detail on pages 182-210. (I haven't the faintest idea why Commodore decided to spell ''Kernal'' the way they did!) These Kernal routines are fairly thorough and cover most of the important input/output type operations. As time goes on, though, you will find that there's more to machine language programming than just input/output type operations. You will need to input numerals, print strings to the screen, reset BASIC and so on, and none of the Kernal routines cover this sort of activity.

Of course, you can always write routines to fulfill these additional functions, but it's always nice to be able use ROM routines when possible. By doing so you will save development time and program space as well. After all, there's no need to reinvent the wheel!

But what is the user to do? As far as I know, there is no guide into the intricacies of the non-Kernal routines. Where can the user turn to for help? In cases like this, there's only one thing to do and that is to band together and help each other. Let's write our own manual!

To get things started, here follows a list of non-Kernal routines that I have found and used recently. All of these have been tested and tried in various programs of one sort or another, and they seem to work quite well. In describing them, I have tried to follow a similar format utilized by the VIC-20 PROGRAMMER'S REFERENCE GUIDE. Thus a function name is given, along with a call address, statement of purpose and a list of the registers affected.

It is hoped that this format will make the list easier to use with the REFERENCE GUIDE, and in fact you may wish to clip these pages and stick them in with your manual.

A few comments should be made about the function names given. There's absolutely nothing sacred about these names; I tried to pick some nice six letter names that suggest the meaning of the subroutine and yet didn't conflict with other names already in use. Six letters is the limitation on labels for my assembler and is a nice round number as well. In the final analysis, of course, the name of a routine isn't that important; it's what it does that counts!

I have used a convention in denoting certain zero page locations in the following descriptions. When talking about a single location, a simple number is used, i.e. $22 means location $22 and nothing more. But when describing a pointer, such as the two byte pointer $22 and $23, then the convetion ($22) is used. Thus ($22) means location $22 and $23; the low byte and high byte, respectively, of the pointer. This is the same convention used by a number of sources in describing 6502 code.

To arrive at the "how-to-do-it" for these routines I started with Jim Butterfield's memory mapfortheVIC-20,("moreVICMaps",COMPUTE!,Volume 4, Number 3, Issue 22, March 1982, pp.168, 170-175) and used a homebrew disassembler to help fill in the details. I also found Raeto Collin West's marvelous tome, PROGRAMMING THE PET/CBM, (Greensboro, North Carolina: COMPUTE! Books in conjuction with Level Limited, 1982) to be of graet value.

I hope this gets the ball rolling on interpreting and using the non-Kernal routines in the Vic-20. This humble list is only a start so I encourage everyone to get into the act and contribute a page to this new "manual" we're **writting! Send in your tips and suggestions to The** TORPET and let's get a whole new world of machine language programming going for the VIC-20.

Function Name: BLKMOV
Purpose: transfer a block of bytes
Call address: $C3BF
Registers affected: A, X, Y

Description: This routine moves a block of bytes from one location to another. If the destination block doesn't overlap with the source block, then the bytes may be moved in either an upward or a downward direction. If there is overlap, then the bytes may only be moved upward.

To use this routine, load ($5F) with the start address of the source block. Load($5A) with the end address+1 of the source block. Then load ($58) with the end address 1 of the destination block and call the block move routine. The block of bytes will be transferred accordingly.

Function Name: WARMST
Purpose: reset BASIC and wait for next command
Call adress: $C474
Registers affected: not applicable

Description: This is a good terminal point for many programs. If after executing certain functions, you wish the VIC-20 to reset the BASIC interpreter and wait for another command, then jump to this address.

This is the graceful exit path; compare this routine with ERROR (see below).

Function Name: CHAIN
Purpose: relink BASIC program lines
Call address: $C533
Registers affected: A, X, Y

Description: CHAIN will go through an entire BASIC program in memory and rebuild the chaining of the lines. Recall that a BASIC line is stored with two bytes devoted to the link address, two bytes for the line number, a variable amount of bytes for the program line and a terminating zero byte. The final line in memory is followed by two additional zero bytes. Thus, the end of a program actually concludes with total of three zero bytes.

There is no setup reguired for this routine. Simply call it and the lines will be relinked. Upon return, location ($22) will be pointing at the second of the three zero bytes mentioned above. This information will sometimes be important to know.

CHAIN is often found in toolkit commands like DELETE and less frequently in assembler/editors.

Function Name: CRUNCH
Purpose:crunch input line into BASIC tokens
Call address: $C57C
Registers affected: A, X, Y

Description: This routine will take a line from the input buffer ($0200), crunch it into BASIC tokens and redeposit it in the buffer. A little reflection should convince you that the crunched line will always be smaller than or equal to the input line, hence this double use of the input buffer is justified.

There is no setup needed for this routine. Just call it, and whatever is currently in the input buffer is crunched. The input line,by the way, should be terminated with a zero byte.

This routine is most frequently used in toolkit commands like FIND and CHANGE. When trying to FIND a BASIC keyword, it must first be crunched so that the VIC-20 will recognize it as a token.

Function Name: SEARCH
Purpose: find the address of a desired line in memory
Call address: $C613
Registers affected: A, X, Y

Description: Location ($14) is loaded with a line number of some BASIC line (low byte, high byte) and then this routine is called. If the routine returns with the carry flag set, then the line was found and ($5F) contains the address of the start of the line. If the carry is clear, then the line was not found.

A very useful feature is as follows. Suppose that the first line in your program is numbered 100 ane you ask this routine to find line 99 (which doesn't exist). Then the carry is cleared, as men-

tioned above, but in addition ($5F) points to the first line beyound the desired one. In this case it would be pointing towards line number 100.

Likewise, suppose that the last line in your program is numbered 1000 and you ask this routine to find line number 1001. The carry is cleared, as before, and ($5F) will point at the last line in the program. In this case it points at line number 1000.

This routine is commonly found in toolkit commands like DELETE and RENUMBER.

Function Name: CLR
Purpose: perform a BASIC CLR
Cal address: $C659
Registers affected: A, X, Y

Description: This routine will have the same effect as typing CLR in BASIC. Among other things, the pointers to strings and arrays are reset, all files are closed and default devices restored, a RESTORE is performed, the stack reset, CONT is disabled, and the FNX/subscript flag is reset.

You will most often use this routine as a terminal point in a routine or program since quite a few things are reset. For example, a typical assembler/editor might call this routine after each line is input and this will insure that the pointers are not all fouled up.

Function Name: INTEGER
Purpose: fetch an integer input
Call address: $C96B
Registers affected: A, X

Description: INTEGR is used in conjunction with the CHRGET routine to accept an integer from the keyboard. You must set up for the routine by aiming the CHRGET pointer ($7A) at the first byte of the ASCII character as well. After these conditions have been met, call this routine and the integer result will be in ($13).

If this routine is called to accept integers greater than 63999 then a syntax error will occur. The routine will be aborted and you will be directed to the VIC-20's normal error handling routine. Thus INTEGR should only be used in programs that are to operate like the BASIC interpreter, since you wouldn't want to be sent to the normal error routine.

INTEGR is a fundamental machine language routine; it is used on nearly every toolkit command, and in countless other programs as well.

Function Name: PSTRNG
Purpose: print a string to the screen
Call address: $CB1E
REgisters affected: A, X, Y

Description: This routine will print an entire message to the screen. The desired ASCII string should reside in memory somewhere with the accumulator containing the low byte of the first character's address and the Y register the high byte. The string should end with a zero byte. After these conditions have been met, PSTRNG may be called, and the message or string will be sent to the screen only. This applies even if various output devices are in operation.

PSTRNG is obviously quite handy for printing warnings, prompts, sign-on messages,etc. While this routine is good for printing messages, PRLINE (to be discussed shortly), is better for printing numers to the screen.

Function Name: EVAL
Purpose: evaluate an input expression
Call addresss: $CD9E
Registers affected: A, X, Y

Description: This is a very complicated routine and is perhaps the heart of the VIC-20's entire number and string inputting abilities. Only a brief description can be given here, so we will limit our scope to inputting and evaluating a floating point number.

Before calling EVAL, set the CHRGET pointer so that it is aiming at the first character of an ASCII string representing a number. In addition load the accumulator with the first character. Now call EVAL. The number will be accepted and deposited in the floating point accumulator. Syntax errors can occur if the number is out of range, division by zero implied, etc.

Note that you can check to see if the input was really a number or not by inspecting location $0D after calling this routine. If this location contains a #$00 then the input was indeed a number; if it's #$80 then the input was a string.

Function Name: ERROR
Purpose: give syntax error message and abort operation
Call address: $CF08
Registers affected: A, X, Y

Description: This is the routine which should be called whenever a syntax error has been detected. The traditional message will be printed to the screen and the normal error process carried out. Essentially, whatever you were doing at the time will be aborted and control returned to the "waiting for a command" state in the BASIC interpreter.

This is the flip side of WARMST, mentioned above. Both are terminal routines in the sense that they are usually the last routines to be jumped to in a main loop of the program. You jump to WARMST if everything went okay and to ERROR if a problem was encountered.

Function Name: CONVRT

Purpose: convert a floating point number to an integer

Call address: $D7F7

Registers affected: A, X, Y

Description: Assuming that floating point accumulator #1 contains a number, when this routine is called, that number will be converted to an integer and the result deposited in ($14). Note that the result also sits in the floating point accumulator as well, at ($64).

An easy way to get an input number into the floating point accumulator to begin with, is to use the routine EVAL, mentioned above.

Function Name: REAL

Purpose: convert an integer to a real number (floating point)

call address: $DC49

Registers affected: A, X, Y

Description: This routine is invariably used in conjuction with the routine called ASCII, to print an integer to the screen. See ASCII, below.

Function name: PRLINE

Call address: $DDCD

Purpose: print a line number to the current output device

Registers affected: A, X

Description: PRLINE will print an integer, representing a line number, to the current output device. Thus it will send an integer to the printer as easily as it would to the screen. The setup is as follows: load the X register with the low byte of the line number and the accumulator with the high byte. Then call PRLINE. If the screen is the output device, then the line number will be printed at the current cursor location.

This routine finds it's way into a great many programs, including the "Universal Wedge", various toolkit ROM's, and the VIC-20's own LIST command.

Function Name: ASCII

Purpose: convert a number to ASCII digits

Call address: $DDDD

Registers affected: A, X, Y

Description: Whenever it is desired to convert a number to its ASCII representation, this routine should be called. It is often used with the routine labeled REAL, mentioned above. Here's how to use the two routines together.

Suppose that you wish to convert an integer to an ASCII string. Store the low byte of the integer in $63 and the high byte in $62. (This is not a misprint; usually the order for 6502 type operations is the other way around, but it isn't in this case).

Next load the X register with a #$90 and set the carry with the SEC instruction. Now call REAL and then call ASCII. The integer will be converted to an ASCII string stored at $0100 on up, with a zero byte termininating the string.

You can print the result if you wish,, or push it into the keyboard buffer, or do any number of other things with it. Toolkit commands like RENUMBER and AUTO often call these two routines.

When You Are Thinking of Buying a Computer

by Mayland Harriman Pt. Arthur Texas

"Ask not what Computer is able to do but WHAT DO I WANT MY COMPUTER TO DO FOR ME????"

Almost everyone who buys a computer buys for a different reason than his friends or the guy down the street. Unless you are buying a personal computer just to play games then your needs and desires are most other people.

In the SNUG Membership I imagine I am one of the few people who almost never uses his computer to play games!!! After owning my CBM 2001 for a year and a half I have perhaps three games that I EVER play and then only once or twice a year.

I bought a computer to do the programs I need and I am more than happy to buy them from Programmers and Software Companies. Writing a program is really not my cup of tea! Going into a program that has been written by someone else and changing it a bit is fine and I do it occasionally, but I really have no desire to program.

As an Astrologer I nearly bought a computer that would only do Astrological work but luckily better sense prevailed. I bought one my wife and I could find expanded uses for in the future. I do a lot of writing and many times do Free Lance writing so I knew I wanted Word Processing. I think writing at the computer is the greatest writing I have ever done .... the ease of editing, the formatting, everything is just so much nicer and so much more fun.

So when you are thinking of buying a computer, let your needs and your desires govern the style, the price and the add ons. You are the one who is going to live with the computer .... be sure it is "friendly" to you and no one else.

PROGRAMMING CONTEST

MORE ON THE TPUG PROGRAM CONTEST

The closing date for the contest is set for March 16, 1983. Have all your entries into any of the executive by midnight, Wednesday March 16. Winners will be announced at the April meeting.

As a result of several questions, a few clarifying notes are added to the rules:

a) Waterloo basic is acceptable.

b) Professional programmers are those who derive a substantial part of their income from programming. If you have simply sold a program or two, but you spend most of your time doing something else, then you are eligible for the contest.

c) A commercial program is one which has been developed for the purpose of sale either to a single buyer on a contract or for sale generally through stores or by mail order. If entered, it is clearly understood that the program is not intended for commercial use.

d) All programs entered not only become the property of TPUG, but they will be placed in the Public Domain for general use.

The rules of the contest are repeated below, be sure to get your entry in as soon as possible.

TPUG Contest Rules

There are two categories, Games and Applications.

1. Entries will be judged on the basis of originality, utility, method and ease of operation, self documentation and presentation. The Judges will also consider the internal documentation, programming technique and logic (is it easy to follow?). Good use of graphics will also be taken into account if appropriate.

2. A primary requirement is that the program WORKS!!! We will not take the time to debug a program that crashes midway through it's performance.

3. If a program requires documentation to explain how it works or how to use it, and it cannot be contained in the body of the program itself either as REM statements, or as PRINT statements, then a separate documentation file may be provided.

4. The primary program code must be in BASIC. Machine Language portions may be used, as long as their use is clearly documented and well documented source code is provided. The use of a machine language utility (eg. a machine language sort subroutine) is permitted without documentation as long as it is in the public domain and the source is named - eg from a TPUG utility disk, and the reason for it's use is explained.

5. Points will be deducted for:
-having to list a program to figure out how to use it.
-crashes caused by syntax errors.
-crashes which require resetting the machine.

6. Programs must be written by the person submitting the entry, they must be original in form (please - no more Star-Trek programs), and they must be non-copyright. Further, they must not have been previously published, nor may they be commercial products.

7. Entrants are restricted to amateur programmers, NO professional may enter. Entrants may not be on the TPUG Board of directors, nor may entrants be on the judging committee. Further, anyone with the name of JIM BUTTERFIELD may NOT submit an entry.

8. Entrants must be paid-up members of TPUG as of the date of entry. Associate members are welcome to submit an entry.

9. All programs submitted become the property of TPUG, and it is expressly implied that all entrants agree that the programs submitted may be included in the Library and may be distributed to members.

10. Programs must be submitted on a 4040 format disk, or on a Vic tape with the entrants name and the words TPUG PROGRAM CONTEST clearly printed on a lable attached to the disk or tape, and clearly included at the front of the program in a REM statement. Disks only will be returned after the contest with a selection of programs received during the contest. Tapes will be returned as received.

11. Programs will be accepted prepared to run on a Vic or on a Commodore 64.

12. The judges decision shall be final, and prizes may or may not be awarded in some categories. At the discretion of the judges, special prizes may be awarded for special merit.

The Non Cover Story

The little man on our cover that looks like Santa Claus without his beard is actually Jack Tramiel, the founder, president, and c.e.o. of Commodore.

We had hoped to have a cover story on Jack Tramiel this month, and started several weeks ago to obtain the information. Our original contact at Commodore Canada shuffled us off to a person reputed to be responsible for p.r. who promised us a package containing among other things a couple of articles published earlier in **Fortune** and the **Financial Times**

Unfortunately, that did not materialize and we were asked to contact two senior executives instead. This resulted in our being referred off to California for two phone calls, one after the other, and from thence off somewhere else in the States to a p.r. firm which did not even return our call.

The local library up here in the boondocks doesn't get **Fortune** until after the highschools are done with them - and in the end, which makes the story very short we got nothing.

The editor did meet Jack Tramiel personally on one occasion. And his son Sam on several others, but there is not much I could report here other than personal anecdotes and rumours. Perhaps we have a writer/researcher out there who would like to do a series of articles on the Tramiels, Commodore, and their history. If so contact the editor so we can commission it.

Converting Programs from PET to 64

by Garry Kiziak

Many owners of the new COMMODORE 64 will have access to a large number of programs written originally for the PET computer. It is natural for these people to ask "What is involved in converting these programs so that they will run on the 64?". This article will attempt to detail some of the steps involved and hopefully make the conversion somewhat easier. I will only be discussing conversions involving 2.0 and 4.0 ROM PETS. Those interested in converting programs from 1.0 ROM PETS should be able to make the additional changes necessary.

In many cases, a PET program will run immediately on a 64. In some cases, a few minor changes will make the program workable. In a few cases, major surgery will be required, and in some instances, unless you are heavily into machine language, the conversion will be impossible. The type of conversion required will depend on the makeup of the original program.

As I said above, some programs will run immediately on the 64. These programs will be written entirely in BASIC and will not make use of the commands POKE, PEEK, WAIT, SYS, and USR. The easiest way to determine if a program falls in this category is to simply load the program into the 64 and run it. If it works, great. Otherwise, read on.

*Note: All BASIC programs for the PET will load into the 64 correctly. This may seem surprising since a PET program is stored in memory starting at location 1025 while 64 programs normally start at 2049. Such loads are successful because of a relocation feature incorporated into the COMMODORE 64 (and also the VIC 20) computer. These computers' will automatically load a program at the START OF BASIC (wherever that happens to be), unless told to do otherwise (see your manual to see how to tell it to do otherwise).

**Editor's note: The converse is not true. i.e. a program saved from a 64 will not load into a PET.

I should point out that I have had some difficulty loading programs that were saved on a PET with 1.0 ROMS. Such programs do list but the first line is usually mangled. This can be fixed up by deleting that first line and retyping it or by a few simple pokes.

THE SIMPLEST CONVERSION

Of the programs that do require conversion, the simplest to fix are the ones that do not use the SYS or USR commands. They may use the POKE, PEEK, or WAIT commands, but these can usually be fixed up by changing an appropriate address and possibly a corresponding numeric value.

For example, POKE 59468,14 is a command frequently found in PET programs to convert the screen display to lower case. If this command is executed on the 64, nothing drastic will happen but lower case is definitely not displayed. The correct command on the 64 is POKE 53272,23. Thus part of the conversion process will be to locate all POKE 59468,14 statements in the program and change them to POKE 53272,23. Similarly, all POKE 59468,12 statements will have to be changed to POKE 53272,21. (This converts the screen display to upper case and graphics).

The majority of "fixes" can be achieved in this manner. i.e.:

i) Find the address on the PET that is causing a problem
ii) Find the corresponding address on the 64.
iii) Make all changes involving that address.

What is needed then is a list of addresses for the PET that can cause problems and a list of the corresponding addresses for the 64.

Actually, with a little more work, we can even do better. Ideally, a program should be able to run on any machine – PET with 2.0 ROMS, PET with 4.0 ROMS, and the COMMODORE 64.

This can be achieved for the upper/lower case conversion above in the following way.

Assume first that the program is running on a PET. Somehow have the computer execute the following commands:

```
3000 TEXT = 59468: REM Address to be
       poked for upper/lower case

3010 UC = 12: REM Value to be
       poked for upper case

3020 LC = 14: REM Value to be
       poked for lower case
```

On the other hand, if the program is running on a 64, have it execute the following:

```
3100 TEXT = 53272
3110 UC = 21
3120 LC = 23
```

Now change all POKE 59468,12 statements to POKE TEXT, UC and all POKE 59468,14 statements to POKE TEXT, LC. After these changes are made, the correct case will be displayed regardless of which computer the program is running on. If all other problem addresses can be fixed up in this manner, then we are well on our way to converting the program to work on all three computers.

WHICH COMPUTER ARE YOU?

The first task then is to somehow identify what type of computer a program is running on.

There already is a standard technique for identifying whether a PET has 2.0 ROMS or 4.0 ROMS; namely,

```
110 IF PEEK (50003) = 160
    THEN ... : REM 4.0 ROMS

120 IF PEEK (50003) = 1
    THEN ... : REM 2.0 ROMS
```

PEEKing location 50003 on a 64 will usually yield a zero. I say "usually" because 50003 is a RAM location on the 64 and is normally unused. However, machine language routines can be placed in that area and so you cannot be 100% sure what location 50003 will contain. The sequence below will get around this problem and will identify the type of computer correctly without destroying any machine code already there.

```
100 X = PEEK (50003):
    POKE 50003,0: Y = PEEK (50003)

110 IF Y = 160 THEN COMP$ = "4.0":
    REM 4.0 ROMS

120 IF Y = 1 THEN COMP$ = "2.0":
    REM 2.0 ROMS

130 IF Y = 0 THEN POKE 50003,X:
    COMP$ = "64":
    REM COMMODORE 64
```

The statement POKE 50003,0 in line 100 has absolutely no effect on 2.0 PETS or 4.0 PETS since location 50003 is in ROM. On the 64, however, it puts a zero into that RAM location. Notice that the original value in location 50003 is saved by the statement X = PEEK(50003) and restored again in line 130 if the computer is identified as being a 64. Note the use of the variable COMP$ to identify the type of computer just in case it is needed again later in the program.

Now the conversion process should be clear. It should include the following:

1) At the beginning of the program, jump to a subroutine that identifies the type of computer that the program is currently running on.

2) In that subroutine, initialize a set of standard variables (such as TEXT, LC, UC, etc.) to the correct values for that computer.

3) Change all references to numerical addresses or values to the corresponding standard variables.

Here is a sample initialization routine.

```
10 GOSUB 60000

20 REM MAIN PROGRAM

60000 X = PEEK(50003):
      POKE 50003,0:Y = PEEK(50003)

60010 REM INITIALIZE VARIABLES
      COMMON TO 2.0 & 4.0 PETS

60020 TEXT = 59468:UC = 12:LC = 14:
     SCREEN = 32768:HIV = 144

60030 NUMCHAR = 158:KEY = 151:
      NOKEY = 255

60040 IF Y1 THEN 60100

60050 REM INITIALIZE VARIABLES
      PECULIAR TO 2.0 PETS

60060 COMP$ = "2.0":ENA = 46:DIS = 49

60070 RETURN

60100 IF Y160 THEN 60200

60110 REM INITIALIZE VARIABLES
      PECULIAR TO 4.0 PETS

60120 COMP$ = "4.0":ENA = 85:DIS = 88

60130 RETURN

60200 IF Y0 THEN 60300

60210 REM INITIALIZE VARIABLES
      PECULIAR TO THE 64

60220 COMP$ = "64":TEXT = 53272:UC = 21:
      LC = 23:SCREEN = 1024:HIV = 788

60230 NUMCHAR = 198:KEY = 203:
      NOKEY = 64:ENA = 49:DIS = 52

60240 POKE 50003,X:RETURN

60300 PRINT " I DON'T RECOGNIZE
      THIS COMPUTER.":END
```

The variables SCREEN, NUMCHAR, etc. will be explained shortly.

MORE PROBLEM AREAS

Upper/lower case conversion is certainly not the only problem area. Another potential one is the screen.

1. THE SCREEN

On the PET the screen is found in memory locations 32768 - 33767. On the 64, it is found in locations 1024 - 2023.

If all output to the screen is obtained through the use of PRINT statements, then absolutely no problem will arise. If however, the output is POKEd to the screen, then changes will be required.

These changes are best achieved by assigning a value to the base address of the screen and then using an appropriate offset from that base.

For example, the base address of the screen on the PET is 32768 while on the 64 it is 1024. Therefore, the first thing to do is to assign values to the standard variable SCREEN as follows:

SCREEN = 32768 if on a PET

SCREEN = 1024 if on a COMMODORE 64

i) Poking a single value onto the screen.

A statement of the form POKE 32956,61 on a PET has to be changed as follows:
First, calculate the offset.

```
Offset = 32956 - 32768
       = 188

        Then, change POKE 32956,61 to:
            POKE SCREEN +188,61
```

The resulting statement will work on either a PET or a 64 (assuming SCREEN has been properly initialized).

Notice that the 61 does not have to be changed as these values are the same for both PETS and the 64.

ii) Poking within a loop.

The following is a typical PET routine that POKES a border of oreversed diamondsd around the screen.

```
100 FOR I 32768 TO 32807:
    POKE I,218 : NEXT

110 FOR I 32847 to 33767 STEP 40:
    POKE I,218 : NEXT

120 FOR I 33766 to 33328 STEP -1:
    POKE I,218 : NEXT

130 FOR I 33688 to 32768 STEP -40:
POKE I,218 : NEXT
```

This can be changed to work on both PETS and 64 by changing each screen address as above.

```
100 FOR I = SCREEN TO SCREEN +39:
    POKE I,218 : NEXT

110 FOR I = SCREEN +79 to SCREEN +999
    STEP 40: POKE I,218 : NEXT

120 FOR I = SCREEN + 998 TO SCREEN +990
     STEP -1: POKE I,218 : NEXT

130 FOR I = SCREEN + 920 TO SCREEN
    STEP -40: POKE I,218 : NEXT
```

Or better yet:

```
100 FOR I  =  0 TO 39:
    POKE SCREEN +I,218 : NEXT

110 FOR I = 1 TO 24:
    POKE SCREEN +39 + I *40,218 : NEXT

120 FOR I  =  38 TO 0 STEP -1:
    POKE SCREEN +960 +I,218 : NEXT

130 FOR I  =  23 TO 1 STEP -1:
    POKE SCREEN +I*40,218 : NEXT
```

2. CLEARING THE KEYBOARD BUFFER

The PET is able to retain up to 10 keystrokes in a buffer, enabling you touch typists to type as fast as you can without losing any keystrokes. This can sometimes add extra unwanted characters to the beginning of an input, so a common technique in PET programming is to clear the keyboard buffer before each input is requested. This can be accomplished in a couple of ways.

```
1) 100 FOR I = 1 to 10 : GET A$ : NEXT
or
2) 100 POKE 158,0
```

The first method will work as is on the 64. The second method must be changed.

On 2.0 and 4.0 PETS, location 158 always contains the number of characters in the keyboard buffer. On the 64 this value is stored in location 198. Thus if we assign values to the standard variable NUMCHAR as follows:

NUMCHAR = 158 if on a PET

NUMCHAR = 198 if on a 64

and change all references to POKE 158,0 to POKE NUMCHAR,0, then the resulting statement will work on both computers.

3. PAUSING UNTIL ANY KEY IS PRESSED

Here again, two techniques are commonly used.

1) 100 GET A$: IF A$ = "" THEN 100 is certainly the simplest and will work on both computers.

2) 100 POKE 158,0:WAIT 158,1:POKE 158,0 is another technique and will have to be changed to 100 POKE NUMCHAR,0 : WAIT NUMCHAR,1 : POKE NUMCHAR,0

4. WHICH KEY IS PRESSED

A common technique used on the PET, especially in games, is to PEEK at location 151 to see if a key is being pressed and if so, which one. Depending on which key is pressed a certain action is performed. This technique is frequently used in games that use the numeric keypad as a joystick. A sample sequence might be:

```
500 X = PEEK (151)

510 IF X = 255 THEN 1000:
     REM NO KEYPRESS

520 IF X = 18 THEN 2000:
     REM 2 KEY IS PRESSED

530 IF X = 50 THEN 3000:
     REM 8 KEY IS PRESSED
```

etc.

The conversion here is a little more complicated but is still possible. First, we need to know that location 151 on the PET corresponds to location 203 on the 64. Then assign the following values to the standard variable KEY:

KEY = 151 if on a PET

KEY = 203 if on a 64

Replacing line 500 with

500 X = PEEK(KEY)

gives us a start with the conversion.

Another problem occurs with the values stored in location 151 (or 203) when a key is not being pressed. Location 151 on the PET contains 255 while location 203 on the 64 contains 64. This time we will use the standard variable NOKEY and initialize it as follows:

NOKEY = 255 if on a PET

NOKEY = 64 if on a 64

Line 510 is then replaced with

```
510 IF X  = NOKEY THEN 1000
```

There are two problems associated with the other keys. First, location 151 will contain a certain value on the 2.0 machines, the same value on the Skinny 40 (i.e. the 9 inch screen) machines, but a different value on the Fat 40 machines. There is no standard way, that I am aware of, for distinguishing between a Skinny 40 and a Fat 40 machine. But PEEKing at location 57344 will do as well as any other. On a Skinny 40 you will get a value of 169 while on a Fat 40 you will get a value of 76.

5. DISABLING THE STOP KEY

The stop key on the PET can be disabled by altering the Hardware Interrupt Vector. For example:

POKE 144,49 for 2.0 PETS

POKE 144,88 for 4.0 PETS

will disable the stop key (and the time clock as well).

The corresponding command on the 64 is

POKE 788,52

To enable the stop key again

POKE 144,46 for 2.0 PETS

POKE 144,85 for 4.0 PETS

and POKE 788,49 for the 64

These can be replaced by

POKE HIV,DIS to disable the stop key

and POKE HIV,ENA to enable the stop key

after appropriately initializing the variables HIV, DIS, and ENA. On the 64, the program can still be stopped by pressing the RUN/STOP and RESTORE keys simultaneously, but this will prevent stoppage of a program due to accidentally pressing the STOP key.

A good question to ask is "How do you know what value is to be stored in these locations?". The PET program actually tells you the location to poke as well as the value, but the value to be poked on the 64 is usually different (c.f. disabling the stop key above or converting to upper/lower case). A memory map will tell you what location to poke on the 64, but it will not tell you what value to poke it with. A good start is to PEEK that location from direct mode and make note of the value. Do this for all three machines and it will tell you the "normal" state of that location. For example, PEEKing location 144 on 2.0 PETS and 4.0 PETS yields 46 and 85 respectively. PEEKing at 788 on the 64 yields 49.

Observing that the value to disable the stop key on the 2.0 and 4.0 PETS are each three more than the ''normal'' value, a good start to finding the correct value on the 64 is to add 3 to the tnormal value of 49, obtaining 52. This process will work for more than 90% of the problem values. It is that last 5 - 10% that makes the conversion challenging.

It would be impossible to list all problem locations and their ''fixes'' here (I will list what I feel are the more common ones below). Instead I have attempted to give you a feeling for how the conversion should proceed. The proper tools that are required are the excellent memory maps (both zero page and ROM routines) published for all three computers in COMPUTE by Jim Butterfield. Another excellent source is the book PROGRAMMING THE PET/CBM by Raeto Collin West, and I am sure there are others. (Would you believe **The TORPET** -ed.).

SOME OF THE MORE COMMON PROBLEM LOCATIONS

Location On 2.0 PET	4.0 PET	64	Name	Suggested Description
40-41	40-41	43-44	SBAS	Start of BASIC text
42-43	42-43	45-46	SVAR	Start of variables
44-45	44-45	47-48	SARR	Start of arrays
46-47	46-47	49-50	EARR	End of arrays
52-53	52-53	55-56	MEM	Top of memory
144-144	144-145	788-789	HIV	Hardware Interrupt Vector
151	151	203	KEY	Which key is pressed
158	158	198	NUMCHAR	Number of characters in keyboard buffer
159	159	199	RVS	Screen reverse flag
167	167	204	CRSR	Flag for flashing cursor in GET statements
196	196	209	SLO	Pointer to screen
197	197	210	SHI	(low/high format)
198	198	211	CH	Horizontal position of cursor
216	216	214	CV	Vertical position of cursor
623	623	631	BUFF	Start of keyboard buffer
634	634	-	-	Start of first cassette buffer
826	826	828	CAS	Start of second cassette buffer
32768	32768	1024	SCREEN	Start of screen memory
59468	59468	53272	TEXT	Poke location for upper/lower case
64721	64790	64738	-	Simulates power on reset

SPECIAL VALUES

12	12	21	UC	Upper case
14	14	23	LC	Lower case
255	255	64	NOKEY	Nokey is pressed
46	85	49	ENA	Enable stop key
49	88	52	DIS	Disable stop key

A COUPLE OF CAUTIONS

1. The PET and the 64 only recognize the first two letters of a variable name. When converting a program you must make certain that the variables already present in the program do not conflict with the standard variables suggested above. If there is a conflict, change whichever you feel is easier.

2. If a program is to be used both on a PET and a 64, then the changes should be made on and saved with a PET computer. The reason for this is that a program saved on a 64 will not load properly on a PET, due to the lack of a relocation feature in the PET, (But see below).

If the changes are made on a PET, then a utility such as BASIC AID or POWER will be invaluable since you can type in such things as

FIND/POKE/

and all lines that contain a POKE statement will be listed, making it easier for you to make the necessary changes and to make certain that you have found all of them.

Similarly you can type in

FIND/SC/

to see if there are any variables in the program that will conflict with the standard variable SCREEN.

PROGRAMS THAT CONTAIN SYS OR USR COMMANDS

These programs will require that you be somewhat familiar with machine language in order for you to be able to make the necessary conversions. Such changes are beyond the scope of this article. However, let me say that these M/L routines themselves fall into a number of categories.

1) The routine works on the 64 as is. (Few routines will likely fall in this category)

2) The routine will work with a simple address change. (These are frequently ROM routines such as the routine for resetting the entire stack.)

3) The routine will work with some minor changes. An example here could be a routine to reverse a portion of the screen. Chances are the only changes necessary would be for the location that determines the base address of the screen. However, if parameters are passed in the calling statement, then the location of certain ROM routines (such as checking for a comma) might have to be changed as well.

4) The routine will require major surgery before it will work. A program like BASIC AID or MICROMON would fall into this category. Such programs should be left to the experienced users.

Other areas that may require major changes are those programs that make use of the BASIC 4.0 disk commands. Some of these can be fixed up easily, but some are extremely difficult (e.g. those that make use of Relative Record files).

Programs that make use of CB2 sound will still run on the 64, but no sound will be produced. The POKEs that the PET uses to produce these sounds will POKE into the ROMS of the 64 and hence do no harm. Once you are familiar with the sound process on the 64, here is a good place to make use of the variable COMP$. For example, suppose lines 1000 - 1030 in the PET program are used to produce the sound. Leave these lines exactly as they are and add a similar routine for producing sound on the 64 beginning with

1031 IF COMP$''64'' THEN 1040

Your sound routine can be placed in lines 1032 - 1039 and the rest of the program should proceed as normal.

LOADING PROGRAMS SAVED ON THE 64 INTO THE PET

As mentioned above, programs saved on the 64 do not load properly into a PET. It is not a difficult procedure to correct this shortcoming however. Here are the steps.

1. Type in a dummy line 0 into the PET, 0 REM will do.
2. Type in POKE 2048,0.
3. LOAD in the program that was saved on the 64
   as you normally would.
4. Type in POKE 1025,1 : POKE 1026,8.

You should now be able to LIST the program including the dummy line 0 that you typed in initially. Delete this line by typing in 0 . The process is now complete. You can save the program to cassette or disk. The next time that you load it into your PET, it will load normally. If you are using a disk, you will notice that the program is 3 blocks longer than the original even though it is the same program. The reason for this is that the Start of Variables pointer did not get changed properly. An experienced programmer can get into the monitor and make the necessary changes without too much difficulty, but the program will operate correctly without making this change.

PET-VIC-64

Cross Reference Map

by Mark Niggeman

These entry points for each of the Commodore ROM sets that are out, represent some of the most called routines in many machine language programs.The programmer is cautioned to check for proper setup of registers, memory locations, etc., before calling any of these routines.

Common Entry Points
Original/Upgrade/4.0 Rom/VIC 20/CBM 64

Compiled by Mark Niggemann

Orig	Upgr	4.0	VIC	64	DESCRIPTION
C357	C355	B3CD	C435	A435	?OUT OF MEMORY
C359	C357	B3CF	C437	A437	Send Basic error message
C38B	C389	B3FF	C474	A474	Warm start, Basic
C3AC	C3AB	B41F	C49C	A49C	Crunch and insert line
C430	C439	B4AD	C52A	A52A	Fix chaing and READY
C433	C422	B4B6	C533	A533	Fix chaing
C48D	C495	B4FB	C579	A579	Crunch tokens
C522	C52C	B5A3	C613	A613	Find Basic line
C533	C55D	B5D4	C642	A642	Do NEW
C567	C572	B5E9	C659	A659	Reset Basic and do CLR
C56A	C575	B5EC	C65E	A65E	Do CLR
C59A	C5A7	B622	C68E	A68E	Reset Basic to start
C6B5	C6C4	B74A	C7AE	A7AE	Continue Basic execution
C863	C873	B8F6	C96B	A96B	Fix point number from Basic
C9CE	C9DE	BADB	CADE	AADE	Send RETURN, line feed
CA27	CA1C	BB1D	CB1E	AB1E	Print string
CA2D	CA22	BB23	CB24	AB24	Print precomputed string
CA47	CA43	BB44	CB45	AB45	Print'?'
CA49	CA45	BB46	CB47	AB47	Print character
CE11	CDF8	BEF5	CEFD	AEFD	Check for coma
CE13	CDFA	BEF7	CEFF	AEFF	Check for specfic character
CE1C	CE03	BF00	CF08	AF08	'SYNTAX ERROR'
CFD7	CFC9	C187	D0E7	B0E7	Find fl-pt variable, given name
D079	D069	C2B9	D185	B185	Bump variable address by 2
D0A7	D09A	C2EA	D1BF	B1BF	Float to fix conversion
D278	D26D	C4BC	D391	B391	Fixed to float conversion
D679	D67B	C8D7	D7A1	B7A1	Get byte to X reg
D68D	D68F	C8BE	D7B5	B7B5	Evaluate string
D6C4	D6C6	C921	D7EB	B7EB	Get two parameters
D73C	D773	C99D	D867	B867	Add (from memory)
D8FD	D934	CB5E	DA28	BA28	Multiply from memory location
D9B4	D9EE	CC18	DAE2	BAD2	Multiply by ten
DA74	DAAE	CCD8	DBA2	BBA2	Unpack memory variable to FAC#1
DAA9	DAE3	CD0D	DBA2	BBA2	Copy FAC#1 to (X,Y) location
DB1B	DB55	CD7F	DC49	BC49	Completion of fixed to float
DC9F	DCD9	CF83	DDCD	BDCD	Print fixed-pt value conversion
DCA9	DCE3	CF8D	DDD7	BDD7	Print floating pt value
DCAF	DCE9	CF93	DDDD	BDDD	Convert number to ascii string
E3EA	E3D8	E202	E742	E716	Print a character
NA	E775	D722	NA	NA	Output byte as 2 hex digits
NA	E7A7	D754	NA	NA	Input 2 hex digits to .A
NA	E7B6	D763	NA	NA	Input 1 hex digit to .A
E7DE	F156	F185	F1E6	F12F	Print system message
F0B6	F0B6	F0D2	FFB4	FFB4	Send 'TALK' to bus
F0BA	F0BA	F0D5	FFB1	FFB1	Send 'LISTEN' to bus
F12C	F128	F143	FF93	FF93	Send 'LISTEN' Secondary Address
F12C	F128	F143	FF96	FF96	Send 'TALK' Secondary Address
F167	F16F	F19E	FFA8	FFA8	Send character to bus
F17A	F17F	F1B6	FFAB	FFAB	Send 'UNTALK' to bus
F17E	F183	F1B9	FFAE	FFAE	Send 'UNLISTEN' to bus
F187	F18C	F1C0	FFA5	FFA5	Input from bus
F2CD	F2AE	F2E2	FFC3	FFC3	Close logical file in .A
F32A	F301	F335	FFE1	FFE1	Check stop key
F33F	F315	F349	F1E2	F12B	Send message if direct
NA	F322	F356	FFD5	FFD5	Load subroutine
F3FF	F40A	F449	F647	F5AF	Print 'SEARCHING'
F411	F41D	F45C	F659	F5C1	Print file name
F43F	F447	F486	NA	NA	Get load/save type parms
F462	F466	F4A5	FFC9	FFC9	Open channel for output
F495	F494	F4D3	F867	F7EA	Find specific tape header block
F52A	F521	F560	FFC0	FFC0	Open logical file
F579	F56E	F5AD	F784	F701	?FILE NOT FOUND
F57B	F570	F5AF	F798	F715	Send error message
F5AE	F5A6	F5E5	F7AF	F72C	Find any tape block header
F667	F656	F695	F84D	F7D0	Set tape buffer start address
F67D	F66C	F6AB	F854	F7D7	Set cassette buffer pointers
F6E6	F6F0	F72F	FFCC	FFCC	Close channel
F78B	F770	F7AF	FFC6	FFC6	Set input device from LFN
F7DC	F7BC	F7DF	FFC9	FFC9	Set output device from LFN
F83B	F812	F857	F894	F817	PRESS PLAY..; wait
F85E	F835	F87A	F8AB	F82E	Sense tape switch
F87F	F855	F89A	F8C0	F841	Read tape to buffer
F88A	F85E	F8A3	F8C9	F849	Read tape
F8B9	F886	F8CB	F8E3	F864	Write tape from buffer
F8C1	F88E	F8D3	F8E8	F869	Write tape, leader length in .A
F913	F8E6	F92B	F94B	F8D0	Wait for I/O complete or stop key
FBDC	FB76	FBBB	FBD2	FB8E	Reset tape I/O pointer
FD1B	FC9B	FCE0	FCF6	FCB8	Set interupt vector
FFC6	FFC6	FFC6	FFC6	FFC6	Set input device
FFC9	FFC9	FFC9	FFC9	FFC9	Set output device
FFCC	FFCC	FFCC	FFCC	FFCC	Restore default I/O
FFCF	FFCF	FFCF	FFCF	FFCF	Input character
FFD2	FFD2	FFD2	FFD2	FFD2	Output character
FFD5	FFD5	FFD5	FFD5	FFD5	Load program to ram
FFD8	FFD8	FFD8	FFD8	FFD8	Save program from ram
FFE1	FFE1	FFE1	FFE1	FFE1	Check stop key
FFE4	FFE4	FFE4	FFE4	FFE4	Get character

Interfacing a Commodore 64 to a Video Monitor AND AN IDS MICROPRISM 480 PRINTER

by John Ason Pittsburgh, Pa.

The Commodore 64 is a powerful personal computer at a reasonable price. However, taking full advantage of the CBM 64's capabilities, via a video monitor and non-Commodore printer, can be frustrating to a novice due to a lack of adequate documentation. Some of the mystique should be removed by the information provided in this article.

If you intend to use the CBM 64 for business related purposes, a video monitor is suggested to provide improved clarity and less eyestrain. The instructions in the Commodore 64 User's Guide do not clearly explain what is necessary in order to interface a monitor to a Commodore 64.

After several discussions with Commodore (via their customer assistance number (215/687-4311 or 4312)), I obtained an instruction sheet for the cable interface. Several local Commodore dealers stated that a cable would be available in the near future for approximately $20 (presently available by mail from ECX Computer Company, 415/944-9277, for $19.95, Part number MX-601).

I chose to make my own cable with a five pin DIN plug (Radio Shack 274-003, $1.49) and a cable with a RCA style jack on one end (Radio Shack42-2371, $1.89). My total cost was $3.37. The cable ground is soldered to Pin 2 and the video wire is soldered to Pin 4 in the DIN plug. Based on my lilited soldering experience, this cable can be made by most novices at a savings of $16.59.

Interfacing the Commodore 64 to my IDS Microprism 480, and apparently, most printers other than Commodore printers, can be a frustrating experience due to the need to convert the CBM 64 Commodore ASCII to standard ASCII for output to a non-Commodore printer. The task is somewhat simplified by the purchase of an interface cable/Commodore ASCII to ASCII converter (available from ECX Computer Company, Part Number CX-6401, $89.95).

I experienced two interface problems with the CX-6401. First, the IDS 480 interface connector requires an input from a Cinch DB-25S ($6.95) or equivalent connector. The CX-6401 output must be to an Amphenol 57-50360 connector (Centronics Compatible - $8.95). The Cinch and Amphenol connectors are not compatible. The solution to this problem is a second interface cable for the Cinch and Amphenol connectors, which is described in Appendix A of the IDS 480 Owner's Manual.

The second problem dealt with the CX-6401 power requirements. The CX-6401 module requires a 5 volt input from the IDS 480. The power was obtained by soldering a wire to a 7400 Series integrated circuit (Pin 14) in the IDS 480. A description of this operation is provided with the CX-6401. The soldering is delicate and not suggested for a novice. The other end of the wire is soldered to the Amphenol 57-50360 in order to provide power to Pin 18 of the CX-6401 Amphenol connector.

Overall, I am pleased with the results of my efforts. The additions to my computer system are quality products providing good results. My thanks to R.S. Siksa who provided the technical assistance necessary to understand the logic of the interfaces and to assemble the cables.

●●●●●●●●●●●●

Copyrighted Articles

The TORPET has received three articles from three different individuals reserving a copyright by the author but giving the TORPET permission to publish the article on a one time basis. Unfortunately, the TORPET cannot publish these articles for the following reasons:

1. The TORPET pays for the articles submitted to it and would therefore expect to receive the copyright, if any, to those articles.

2. It has been the practice of the TORPET to allow anyone to copy its material for distribution in other publications. This has previously been the policy of the MIDNITE GAZZETTE, and is the policy of CHUG HARDCOPY, and so far as we can tell of WINePEG and other similar publications that we receive, so we wish to reciprocate.

3. The continuing TORPET practice, at this time anyway, will be that articles and programs published by it will be placed in the public domain.

●●●●●●●●●

"TOMORROW, WE'RE TRADING
THIS IN FOR A VIC-20"

- from CHUG Hardcopy

Commodore Service

You Can't Afford To Be Down!

MISCOE DATA Service
Keeps You
Up and Running

Prompt, professional service available on:

- **Commodore micro-computers**
- **Commodore disk drives**
- **printers**

Several service plans available —
Choose the one best for you.

Third Party Service Specialists Since 1969

MISCOE DATA

Toronto: (416) 677-2745
Montreal: (514) 631-4381

VORTEX

The enemy lurks in swirling rings of energy, your misson is to penetrate his force field and destroy him. The graphic animation alone is worth the price of the game. As usual, our machine language routines create a fast action scenario.

$17.95 cassette

PHANTOM FIGHTER

Against a moving backdrop of mountains shoot down UFOs that appear in ever increasing numbers on the screen. Skill levels allow you to change the flight speed of your fighter plane. Avoid the mountains and clean the screen for high scores, bonus points are given when you reach high point levels.

$17.95 cassette

FROG!

The original version that has been acclaimed by kids af all ages the world over!! Control the height of the frog's jump along with the length of his tongue to score the bugs he eats. Superfrog is a slightly different version of the same game but with a few twists! An oldtime favorite that is a must for every game library.

$6.95 cassette

WASPS

The wind sways the tree branches and angry insects attack you as you try to destroy the wasp nest with poison filled balloons. The air fills with the insects whose stings are fatal. Protect yourself and destroy the nest to score, the more nests you destroy the higher your score.

$11.95 cassette

WEATHER WAR

A two player game that uses natures own weapons of hail rain, tornado and lightning to destroy your opponents house. Prevailing winds effect the accuracy of your shots and a random act of nature can change the outcome of the game just as you think it's all over. Excellent visual display.

$6.95 cassette

AEROSOL

Part of a bug series, Aerosol lets you kill the pests with a blast of your insecticide. Try for as many as possible before the pressure in your spray can runs out. Just like in real life killing the little pests is not as easy as it looks!!

$6.95 cassette

GRAPHIC DUO

Two terrific animated cartoons, Rain Dancer and Petdromeda, show the graphic capabilites of the Pet at its very best. Clever demonstrations of just what you and your Pet can do.

$6.95 caassette

MAGIC CARPET SOFTWARE
P O Box 35115
Phoenix, AZ 85069

PET Disk !!!

All 7 Games ($75.65 value)

for

$49.95

MAGIC CARPET

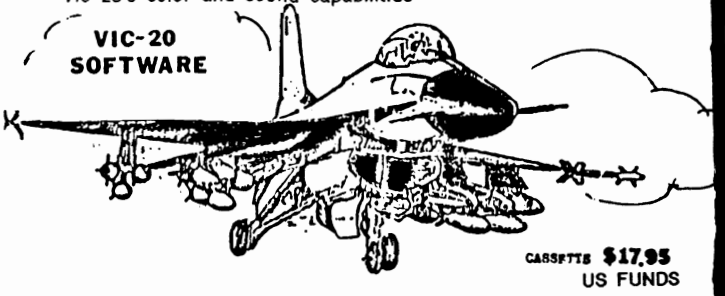
P.O. Box 35115 Phoenix, AZ 85069

VIC-20 SOFTWARE

Phantom Fighter

The Superstition Mountains have become infested with an unearthly menace!! Armed with rockets and bombs, the mission of your supersonic fighter is to clear the slopes of the alien presence. Your aerial combat takes place against a moving backdrop of hazardous mountain terrain.

Phantom has machine language routines to provide high speed animation. This arcade game takes full advantage of Vic-20's color and sound capabilities



CASSETTE **$17.95**
US FUNDS

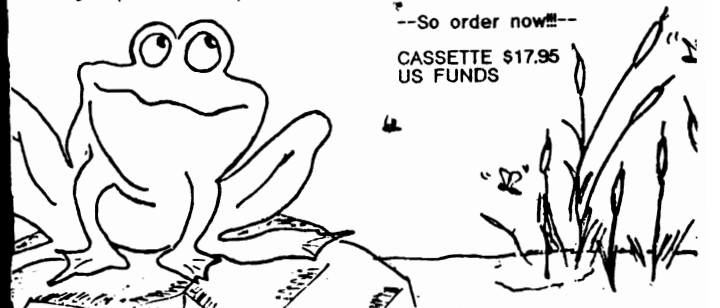
VIC-20 **FROG** SOFTWARE

The guiet of a summers night is broken by the soft flutter of insect wings. From his stony perch, a hungry toad sends his raucous call over the bog.

One of the most popular games ever written for the Pet/CBM now makes its debut on the Vic-20. Creative use of color, superb animation and terrific sound effects makes Frog a favorite in everyone's game library . The game requires no memory expansion or special controllers, just your basic Vic-20.

--So order now!!--

CASSETTE $17.95
US FUNDS



Take control of an animated fly zapper!!

Butterfield Box

by Jim Butterfield

Toronto, Canada

Some Serious Math

Don't worry - I'm not going to bury you in formulas (formulae?) I'd like to talk about a couple of branches of applied mathematics that fits very well with computers. In fact, you won't need to know the math ... just how to use the results.

It's difficult, and perhaps dangerous, to pass out a mathematical tool and say, "Go ahead and use it". Most applications require more than just math: they take insight into what the numbers mean and what your objective are. Yes .. there are quite a few computer users out there who don't know their own objectives.

But if you don't know about the existence of a computer tool, you can't even think about using it. I'd like to mention a few ... briefly.

Linear Programming.

"Linear Programming" has nothing to do with ordinary programming. It's part of a branch of math called "finite mathematics"; it has its own jargon ("convex sets"; "branch and bound"; "simplex method"). It seems confusing.

Yet it isn't. Linear programming deals with either: how to use limited resources, or how to find the most economical combination of items.

For example: you have a small farm, and can raise pigs, sheep, cows and chickens. But you have limitations; only so much acreage for grazing, only so many feed stations, only a certain number of pens or shelters. Each animal takes a different amount of these resources; and each animal has a different profit level. What mix of animals will make the most money?

Another example: you still have the farm, and must feed your hogs from a dozen supply sources. Each type of food has a different content of vitamins, minerals, protein and calories; and you must feed your animals a sufficient quantity of each of the vital ingredients. As prices vary for the different feeds, how can you find the most economical way to keep your pigs healthy and growing?

In both cases, Linear Programming will provide the answer. You don't really have to know how it works; just set up the formulas and start the program.

By the way, a simple but useful Linear Programming program can be found in **Some Common Basic Programs**, published by Osborne/McGraw Hill. But if you want to delve into how it works, look through mathematics texts; if you have the inclination, it can be fascinating reading.

Simulation.

"Simulation" does not mean doing your favorite W.C. Fields imitation. In the computer field, it has a special meaning: working through a hypothetic series of events, usually to spot waiting (or (queueingc) problems.

Suppose you have a supermarket with a certain number of shopping carts and a series of checkout counters. You expect customers to arrive at a certain average rate, take a cart, spend a typical amount of time picking up items, and then head for the checkout, where they will spend a variable amount of time getting their groceries rung up. How can you tell - before you build the store - what the capacity will be? Will customers be turned away because there are no carts, or will the wait in the checkout line be excessive?

Simulation allows you to run the hypothetical supermarket, and let the hypothetical customers wait in hypothetical lines. Using a simulation program, you can try to spot the bottlenecks and fix them before you build the store.

Simple simulation programs can be easily written on a home computer. Let's suppose that we have an automatic bank teller system. At the busy time, users arrive at the system at a rate of one per thirty seconds. The average user spends twenty seconds performing a transaction. No waiting, right?

Wrong. Because arrivals are random, and usage times are random, there will be a wait for many customers (in fact, about two thirds will wait). Let's program it:

```
100 PRINT ''ARR ED'',''SERVED'', ''WAIT''
110 FOR C = 1 TO 100 (100 customrs)
120 A = A-30*LOG(-RND(1))
130 S = A:IF T>S THEN S = T
140 W = S-A
150 T = S-20*LOG(-RND(1))
160 PRINT INT(A);INT(S);INT(W)
170 W0 = W0+W
180 NEXT C
190 PRINT ''AVERAGE WAIT = '';W0/100
```

A few notes: A is the arrival time of the next customer. S is the time that customer starts to get service, and T is the time that the customer is finished. The customer will get service right away if the previous customer has already finished (see line 130); otherwise he or she will get on when the machine is available. Random time is calculated quite oddly - lines 120 and 150 seem to be subtracting, but they are adding a value with a special sexponential scattering.

Watch the program run; you'll see queues formandthendieaway.Somecustomersmightwait quite a long time. We've set an average customer arrival of one per 30 seconds; 100 customers should take 3000 seconds, but this won't work out exactly because of the random factor. This program is written so that it will fit on a VIC screen; if you have more space, or want to use the printer, you can also print T (time finished) and T-S (time on the teller unit).

Summary

There's more to math that just obscure symbols on a blackboard. Learn about the mathematical tools that can be useful to you.

The Hardware Hacker

by Hank Mroczkowski From CHUG Hardcopy

Cheap PET 2001-8 Memory

ZOT! A thought struck me while contemplating the problem of expanding the old PET 2001-8 memory. Owners of that old beastie take note ... this is cheap (?) and easy. First, get the VIC memory expansion module of your choice ... 8K or 16K. Next, you'll need the double-40 pin (0.1'') card connector and one or two double-22 pin (0.156'') card connectors (it depends whether you'll want to go the full 32K) and about a foot of 32 (or more) conductor ribbon cable. Using the pinout of the VIC-20 and of the PET 2001-8 computers, wire the harness to accomodate the cartridges.

I will eventually need to do this when I put my old '2001 board into semi-retirement. I will use the 1st Mate expansion on a 32K PET to get either 64K or upgrade all the printed circuit board to get a total of 256K. That, however, is a ''back burner project'' until the memory management IC's become more common ...6 to 12 months? (Want to bet on it?) It looks to be a very straightforward project, even to the point of not needing buffering ...CMOS is wonderful, isn't it?

When I build mine, I'll give the tested and proven connector wiring diagram. For those who can't wait, talk to your local wizard, dealer (who has a shop) or to Ron Gould and try to work something out. (Ron, by the way is your typical wizard, gone commercial ...so don't expect your high pressure sales pitches ...if it can't be done, he'll say so ...if it can, it will.)

EPROM Holders

The VIC-20 memory cards, adapted to the PET can also hold EPROMS. This looks like you can add the Toolkit(R), Basic-Aid, Micromon, etc. without having to buy those expensive, single-function expansion sockets. Later, you can use the modules on a VIC-20 or sell them to recoup some of your investment ...a good deal all the way around.

Check that older VIC

THe HOTS in your older unrevised VIC's may have dried out your electrolytic capacitor by now. So, owners of the older, version ''E'' take note ...if your picture has hum bars creeping vertically across the picture, the filter capacitor may be going bad. Replace it with one of equal or higher value in both voltage and capacitance. Be sure it fits and be sure you install it in the proper polarity ...they do explode if the voltage is reversed! Get help: if you must.

ALARMING NEWS

ALARMING news ...as cheap as the VIC-20 is, some hackers are begining to look into using their machines to monitor the status of their houses. It's a natural! With all the I/O (that's Input/Output, people) built right into the thing, adapting it to ''real world'' sensing and control is easy. The sound generator can dial the Touch-Tones(R) we all love. The POTX and POTY inputs can sense heat, cold, light, wind, sound and even blood flow. The joystick and the user port lines can detect switches, outputs from ultrasonic alarms, window tape, etc. Output can also be gotten from the user port, the cassette motor switch, the serial (DIN) bus and the video/sound connector. Don't forget to extend the necessary lines from the keyboard to allow yourself a combination keypad.

Superspeed Sort

by David Williams Toronto, Ontario

Several TPUG members have recently contacted me to ask questions about my program "Superspeed Sort" which is on the September 1982 club disk. Here are a few questions and answers about this program.

Q. What does the program do?

A. It is a machine-language "subscript sort, keyed to a string array. If the string array is named AR$() and the subscript array is named SB%(), it will sort the numbers in the subscript array so that the loop:

```
FOR I=0 to N: PRINDT AR$(SB%(I)): NEXT I
```

will print out the strings in ascending alphabetical order. The string array is NOT changed by the sort.

Q. What machines will it run on?

A. All PETs and CBMs with 2.0 (upgrade) or 4.0 BASIC. I imagine it could be modified easily to run on any 6502 (or 6502-compatible) machine with Microsoft BASIC. If anyone modifies it to run on a VIC-20, Commodore 64, or any other machine, please publish the modifications.

Q. What are the advantages of subscript sorting?

A. It is a convenient way of "keysorting" (i.e. accessing other arrays in the order which corresponds to the alphabetical order of an array which has been used as the "key" for the sort). Subscript sorting also allows several different keysorted orders of a set of arrays to be easily stored in the computer's memory. All you need is several subscript arrays, one for each order.

Q. What does the program do if the key string array is not completely filled.

A. It puts (or leaves) the subscripts of the null strings AFTER those of all the filled strings. This seems to me to be the most convenient way of handling partly-filled arrays.

Q. What sort algorithm is used?

A. It is an alternating-direction bubble sort. This algorithm is especially good for MAINTAINING arrays in sorted order. (i.e. for frequently re-sorting them after minor additions, deletions or changes have been made). If the strings are in random order, this algorithm is not as fast as some others, but it is still much faster than anything that could be written in BASIC.

Q. Can I change the names of the subscript array and the key string array?

A. Yes. Instructions for doing this are included in the program. However, the subscript array must be an integer array (using a real-number array would make the sort slower, and would occupy much more memory) and the key string array must be a one-dimensional array. I decided that allowing multi-dimensional arrays would amount to opening too large a can of worms.

Q. Must the arrays be the first ones defined by BASIC?

A. No. The program searches the array table to find the arrays with the right names, so there are no restrictions on the order in which the arrays are set up. If you accidentally try to use arrays which have not yet been DIMensioned, the sort will simply abort, without doing anything awful!

Q. What are the memory requirements of the program?

A. The machine language is about 430 bytes long. The program on the disk includes a BASIC relocating loader which puts the machine language at the top of available memory and protects it with the top-of-memory pointer. The sort uses some zero-page locations as workspace. However, the original contents of these locations are placed on the stack and restored at the end of the sort, so no permanent changes to the zero-page are made. The cassette buffers are not used in any way, so they remain available for other machine language.

Q. Is the program copyrighted?

A. No. However, if you incorporate the sort in any software which will receive wide distribution, I would appreciate my name being included with it.

Q. If I have any other questions, can I contact you?

A. Certainly. My phone number is (416) 486-6817.

TORPET Information

REPORTERS WANTED

Reporters to cover special assignments. Are you attending a convention, going on a trip, visiting a factory, or manufacturer? Are you willing to review a product or take on a special assignment? Whatever your story idea, please contact us ahead of time so we won't overlap on stories.

SUBMIT YOUR ARTICLES NOW!

TORPET BACKISSUES

Backissues of the TORPET are available for $2.00 each (except for issues #1, #2, and #3 which are $1.00).

Issue	pages	Issue	pages
1	4	9	48
2	4	10	32
3	4	11	48
4	8	12	48
5	16	13	24
6	32	14	40
7	48	15	96
8	48		

If you wish to order any of these old TORPETs, please send your cheque or money order to:

TORPET
P.O. Box 100
Station "S" Toronto, Ontario
Canada M5M 4L6

HOW TO SUBMIT ARTICLES

The TORPET is always in need of good articles about the PET, CBM, VIC, C-64, and other related products, software, and subjects. If you wish to submit an article, send it to:

The TORPET Editor
P.O. Box 100
Station "S" Toronto, Ontario
Canada M5M 4L6

To encourage you, we are paying $20 per page that the article fills in the TORPET. If you feel that you have an exceptional article that might command more elsewhere but would like to submit it to us, please do so and tell us what you feel would be the proper remuneration. If the amount is within our budget we may be still be willing to print the article.

Many authors prefer to have their articles printed in the TORPET because it is has the largest completely independent (and completely Commodore) circulation of any magazine in North America. Our press time is usually also much more timely than other magazines.

If you can send your article on disk, it will save us time in re-entering the article ourselves. However, we can also take typed or printed articles or even handwritten ones if necessary. If you send your article in Wordpro, Wordcraft, Final Word, Paperclip, RTC, or many other wordprocessor's format format ON DISK, we will return that disk with the contents of any TPUG library disk of your choice.

The Lowly Cassette

by Mayland Harriman

Pt. Arthur, Texas

Are you embarrassed because you only have cassette for programs? Does it seem like everyone else has dual discs and looks down on you being backwards? Do you get a feeling of inferiority when you pick up a computer magazine or newsletter and the writers are ganging up to make you feel bad when they write that cassettes and tapes are no good and that they predict that most computer stores will not even carry them in the near future?

Take heart old tape friend, you need to be aware of a few things that may cause you to look the bad-mouthers in the eye and tell them they don't know what they are talking about.

The cassette method of storage is the lowest cost way to keep your programs from the initial purchase price to the end results in running your programs. The tape is much slower than disc and it is harder to SAVE a program and DATA together when needed to be stored that way....but remember many of Commodore's computers are designed to use TWO CASSETTES and that narrows the field of objections a little bit more.

There are several programs designed to find a program quickly on a tape and each one I get does the job faster. But better than fast locating is a new chip that I have ordered for my CBM 2001 and which is available for the VIC as well. The chip called the ROM RABBIT allows you to load a 8000K program in about 30 seconds!! Doesn't that sound good? The chip also gives you 12 commands, allows every key on the keyboard to repeat and a few other goodies.

Someday, IF I have many complicated programs with lots of data to save and IF my time becomes much more valuable than it is now, I might go to disc....but remember one disc is considered not enough, you have to have two and that is MONEY, MONEY, MONEY, which I don't care to spend.

Yes, the cost of ADD-ONS are coming down but Disc Drive prices would really have to drop to rock bottom before I can justify the expense. It isn't going to hurt me to type LOAD and go get a cup of coffee or read a few paragraphs in a magazine or something while my CASSETTE does it's job....of course with my new ROM RABBIT chip I will load the programs that I use most of the time in about 45 seconds! I can live with that!

RTC Payroll Package

by Stephen Smith Richmond Hill, Ontario

More and more companies are finding that with the increasingly complex tax laws, they are wasting an ever increasing number of man-hours on calculating weekly payroll checks. For a company with thirty or forty employees it can take up to half a day or more in extreme cases for one employee to write up the checks. If you add to this the time that it takes to go through these figures and add up the total Canadian Pension Plan deductions (CPP) and Unemployment Insurance deductions (UIC) that is due the government from the past month and the amount of time that it takes to fill out T4 slips at the end of the year you will see that it is well worth the effort to look seriously at a computerized payroll package that will make this task a little easier.

The RTC Payroll Package is such a program. It is a general purpose payroll package that makes the Payroll calculations for a company much easier. It was worked on by three people ,Chris Shaw who left RTC to go to university, David Foster worked on it for eight weeks during the summer of 1981, and then myself. It has been worked into a general purpose program that will take into account almost any situation conceivable in a company's payroll tasks.

At present it operates on an 8032 computer with an 8050 disk drive and an 8023p or MX-80 printer. Other versions may be made available at a later date if there is any demand for them. It stores the names, addresses, wages, and tax information for up to forty active employees per data disk. All the information concerning their weekly pay is kept for fifty two weeks at the end of which this package will print up T4 slips with this information.

There are four major parts to the program - all easily accesible from the main menu with self-explanitory prompts. The first deals with personal files. These can be created and edited easily and contain the person's name, address, age, S.I.N., hourly rate or weekly salary, and their tax exemptions as stated on their TD1 forms. There is room for forty employees per data disk in the active files (data disks can be changed as simply as putting in a new disk if you have more than forty employees). There is also room for more in the inactive files for more employees who are temporarily on vacation. This information can be entered, viewed, edited or printed easily by simply selecting the proper option.

The section for adding and editing weekly payroll records or past payroll records is another main section of the program. For entering a week's figures one simply selects the proper option. It calculates the weekly payroll by department and gives a hard copy printout while it is being done. The person's salary or hourly rate is stored in their files and so it merely asks you for the number of hours worked and what benifits or allowances are to be made.

The program automatically calculates the Unemployment Insurance premiums using the tax figures that you have entered when the disk is formatted (in this way, whenever the tax figures change, you simply enter the new ones for your province), and the Canada Pension Plan deductions for you as well as the income tax owing.

A running account is kept of how much vacation pay is owing (usually four percent depending on which option you chose) and this can be deducted whenever it is necessary. These payroll records can be edited if necessary and past records can be added for weeks in that year in which the payroll package was not used so that these figures are present when you want to print up yearly totals or print T4 slips. Subtotals for an employee or for all employees can be viewed . It is simply a matter of entering the figures of what was deducted for that week.

The print payroll or personnel files option allows you to print up monthly summaries with the total remittances for that month for UIC, CPP and tax. You can also print up a monthly breakdown of any preselected deductions that your company has (e.g. a company health plan). These deductions are set up when you format the disk and can be taxable or tax free. You can also print out payroll records for a given week for any specific department or job.

There are options for printing past payrolls for any week in the year and one for printing a weeks cheques. One can either have the cheques printed on printer paper or by defining the positions for the person's name, amount to be paid, etc. Thus you only have to sign the cheques when it is finished.

At the end of the year this package will print up all the necessary information on T4 slips for your employees an prepare a new data disk for the new year. This option will transfer all the active employees to a new data disk and set it up for the next year. Also included in the program are all the necessary commands to change data disks, back them up and format new ones.

Overall the RTC Payroll Package (available from Richvale Telecommunications at 10610 Bayview Ave., Richmond Hill, Ontario) will satisfy the needs of any company's weekly payroll calculations and organization. For the trouble that many company's payroll costs it is well worth the cost to seriously look into this new package.

The RTC Business Systems store and Marc Swanson, manual writer at RTC.

MEETING REPORTS

by John Easton

TPUG WEST Meeting Report
Thursday, 20 January, '83

Hey, and we thought we'd catch a few of you folks with the Thursday meeting night! Well almost everyone still seemed to turn out for the TPUG WEST meeting at Sheridan - only two folks phoned the following Thursday wondering where the meeting was.

Anyhow, this is going to be a quickie, first because a major part of the meeting was taken up by Chris Bennett with a very complete (with coloured slides yet) report on all the new goodies arriving from the Commodore - and I'm certain he has already submitted his script to our illustrious Editor for inclusion in this issue. And second, it's late at night (as usual) and Muriel my wife has suggested rather pointedly that there ARE other things to do of a winter's night than sit at a @!''#$%'& keyboard (wonder what THAT will do to our editor's typesetter!)

First the announcements -

Sheridan College is planning two seminars of hands-on instruction on 6502 Machine-Language. These two-day sessions are scheduled for March 21,22 and March 23,24. (Forget this, they are sold out. However, another seession is in the planning stage for June. -ed.). If you can't make the remaining TPUG West meetings this year, one of these intensive seminars might just be what you're looking for should plain BASIC be proving a bit ho-hum to you. For further information contact the School of Computer Studies, Sheridan College, 845-9430, ext. 142 or 377. The cost - $150.00 (including lunch mind you). (You might check about getting into Jim Butterfield's seminar at the Toronto TPUG May conference also. It is aa freebie. -ed)

Al Farquharson added a few more goodies to the upcoming mid-May Seminar, Faire, Copy session, what-have-you (it is a conference -ed) with mention of the committee's plans to include a Swap/Flea Market type of area to the already gala affair. Actually, Al was asking for volunteers with equipment to lend for one or two of the days involved to PLEASE let him know so that we can spread the load of necessary hardware equally among the members. If you didn't fill in an information sheet (particularly the equipment and/or time portion) I'm certain Al will accept your late submission with gratitude.

And we had a long overdue visit from our hardworking TPUG TAPE department man, Mr. Greenjeans Himself, Peter Smith of Richvale Telecommunications (recently renown for their highly successful V-Link for the VIC and now the 'C64). Peter brought us all up to date by means of stories and anecdotes in his own inimitable style (not quite like Stovekin - but just as difficult to shut up) on the trials and tribulations of being an entrepreneur in this exploding marketplace. Keep it up Peter, you're doing fine.

Incidentally, because you were all so attentive, Peter has promised to come back to future meetings with a small bunch of the stuff you seem to want from him, namely those hard-to-come-by TPUG MONTHLY TAPES.

Should you be interested, Peter had a few copies of an article by Garry Kiziak written for the Canadian Commodore Educational Group (whoever THEY are) on **Converting PET programs for the Commodore 64.** (It is in this issue of The TORPET -ed). Excellent stuff if you're like me and just don't quite know where to start short of buying a '64 and jumping in. Garry (whoever YOU are) please feel free to contact me (yeh, I know, whoever I am!) any time about presenting a few of these goodies at the Westside Meeting.

It's this sort of informative sessions the Westside Group have come to expect and appreciate. In the meanwhile, should you just not be able to wait, Peter has copies available for the very nominal cost of $1.00 - and where else can you get 19 pages photocopied for that price? Come to think of it, the postage alone is 65 cents.... why not send Peter $2.00 - see his address elsewhere in this issue.

Well, with that out of the way, and Muffins and coffee in unlimited supply, we just barely managed to get out on time - not because John Stovekin was talking (where WERE you anyway John?), but because Dave Williams started us all on a tour of the memory to be found in these little beasts. And you thought all the time it was all done with mirrors and little black chips! David, I can't get this all in here in one paragraph - and Muriel has just announced the hour for the LAST time.

A M.L. Presentation

by John Easton

at TPUG West

Anyhow, it all starts out with a simple look at the M/L Monitor (note Dave didn't once mention those terrifying words Machine Language). A simple SYS 4 will get you into this mysterious world of IRQ's, Registers, and Stack Pointers. But not-to worry, (well that's what DAVID said), he merely had us poke around in memory for a look at what might be there.

Since we'd carefully cleared all memory, of course there wasn't much there, not at the Start of Basic anyway where we first looked by entering .M 0400 0440 where the cursor seemed to be asking for something. Quick as a wink, the screen fills with 00 00 00 and a whole bunch of AA's - 45 of them, complete with an associated address for every 8'th number - thoughtfully placed there for those of us who can't count. Like I said, magic.

Why the 3 sets of zero's? That's Basic's way of noting the END OF BASIC, and since BASIC RAM starts at 1024 where we looked, with the request for Memory from 0400 (1024 in hex notation = 0*4096 + 4*256 + 0*16 + 0*1), and since we hadn't yet entered anything that's where the end of basic was noted. Why the AA's? I don't know what the machine calls them, but I do know that empty memory is so designated - how about a mneumonic like All Absent?

So, let's put something into memory with a short one-liner. But WAIT, first we must leave the MONITOR mode by entering our next M/L magic word, .X at the waiting cursor position (and one doesn't really enter the '.' - that's there already) and there we are back in BASIC mode.

Let's try something simple like 10 A= 15.

Can't get much simpler than that now, can we. Now, a quick jump to the MONITOR shows us a Real Live Line of Basic Code in funny numbers!!

```
040000  0A  04  0A  00  41  B2  31
040835  00  00  00  AA  AA  AA  AA
0410AA  AA  AA  AA  AA  AA  AA  AA
etc.
```

So let's examine this a little closer.

0400 - 00, normally 0 value at this location

0401 - 0A

0402 - 04, these two locations point to the start of the NEXT LINE of BASIC and are written in REVERSE Notation, Lo/Hi value (translate to decimal as 040A or 1034)

0403 - 0A

0404 - 00, these two locations take care of the line number 10 (reverse order value again)

0405 - 41 this location indicates our first variable A --- 41 ??? David tells us that a clue to notation of Alpha Characters is indicated by the preface 4 , well, starting at 4 1 for A and progressing upwards into 5 something. I suppose that 4 is really indicative of a decimal value of 64 - from which one would conclude that the value of hex 41 is decimal 65 - which just happens to be the ASCII value of the letter A - and I thought it was magic.

0406 - B2, now there's a funny one, but every time I indicate an = sign that's what the machine thinks I've told it. There must be tables of these values all over the place, but what better way than to ask the machine itself!

0407 - 31

0408 - 35, well what have we left from our BASIC line but the number 15 - and using David's quick clues to NUMERIC notation, one will always find a plain number prefaced by a 3 (not to be confused with funny things like floating point etc. etc.). Anyhow, because hex 31 happens to translate to decimal 49 which is ASCII for the number 1, its plain as day that those two figures represent our number 15 from Basic.

0409 - 00

040A - 00

040B - 00, again our END OF BASIC flag.

Should we care to add another line to our Basic Code, it would begin at 040A (as the notations at 0401 and 0402 have already told us) leaving a single

00 marking the END OF the first LINE at 0409.

Got it so far? Didn't hurt a bit, did it? OK, now that we're so facile in accessing the MONITOR, let's look at the start of SCREEN MEMORY which starts at hex 8000. Access these Memory locations with .M 8000 8XXX (XXX being anything else up to FFF – which would give you all 4096 locations – a wee bit too much for any one screen at a time).

The locations you might want to see are only the first 1000, or in the case of an 80 column machine the first 2000. What about the rest? How should I know, but I do seem to recall something about a duplicate image – why not look around yourself and see, anyhow, let's just access a workable chunk similar to the last time with our upper limit set somewhere around 040.

Unless your screen is absolutely clear at the top, you're about to see a bunch of various numbers, corresponding with whatever character or symbol is at present sitting in that position on the screen – starting in the upper left corner at hex 8000. Incidentally, blanks are represented not by 00 or AA as our previous explorations would lead us to believe, but a 20 (which is hex for ASCII 32 or space.

Now, why not work backwards, change the MONITOR – and see what happens on the screen. Change the contents of location $8000 for instance, by directing your cursor to it's location, changing the contents, and hit return (just like BASIC, isn't it?). Immediately you should see something appear in the 'home' position of your screen.

By something, I mean that if you had been paying attention in the previous paragraphs, you'd naturally be lead to believe that an A might logically be designated by the value 41. Not so – and here we run into what some might call PETASCII.

Commodore machines have a special set of code to address the screen – you've seen it listed in tables perhaps as SCREEN POKE values. Limited in number by the two character limit on individual memory locations to a value not greater than 127 (hex FF), Commodore has scrunched the code tables to allow inclusion of all their fancy graphics and Reverse within this FF limitation. So, thus armed, play around and see what happens (A, incidentally is 01, while reverse A is 81).

David did proceed on into a quick look at useful 0-page (that's up to hex 03FF) addresses, but that's when my pencil broke. We'll leave you here sports fans, – back with more next month, (trusting Dave will pick up where I left off), or better still, why not come out yourself to TPUG West and I can skip this reporting business.

Cheers –
John (the one with the beard, Garry) Easton.

TPUG 1983 Schedule

Central Chapter
Meetings are held at 7:30
at Leaside Public Highschool
Bayview & Eglinton Avenues

Wed.	**Feb 9**	**1983**	
Wed.	**Mar 9**	**1983**	
Wed.	**Apr 12**	**1983**	
Wed.	**June 8**	**1983**	**(last meeting)**

PET Conference
May 14,15, 1983
at George Brown College

Westside Chapter
Meetings are held in the cafeteria
at 7:00
at Sheridan College, Oakville
on Trafalgar Road
(2 miles north of the Q.E.W.)

Wed.	**Feb 23**	**1983**	
Wed.	**Mar 16**	**1983**	
Wed.	**Apr 18**	**1983**	
Wed.	**May 18**	**1983**	
Wed.	**Jun 15**	**1983**	**(last meeting)**

New TPUG Office

The new TPUG office located just south of the 401 on Avenue Road in Toronto is being readied for occupancy. As of printing time for the TORPET it still required painting and furnishing.

Hopefully by next issue we will be able to announce that it will be available for TPUG members to drop off programs for addition to the Library, and articles for the TORPET.

HOW TO SUBMIT PROGRAMS TO TPUG

Programs can be sent to TPUG on either disk or tape. The disk/tape will be returned to you as long as you have enclosed your name and address. It is also a good idea to put your membership number on the tape/disk just in case we misplace the letter or envelope in which it came.

Send all programs to:

TORONTO PET USERS GROUP
P.O. Box 100
Station "S" Toronto, Ontario
Canada M5M 4L6

TPUG FEES

The TPUG fees are paid on an annual basis. This means that if you join in February of 1982, your membership for next year will be due at the END of February of 1983. This is going to help us at renewal time since all the members will not become due at the same time as they did in September last year. TPUG membership always **includes a TORPET subscription** and gives you admission to the annual conference, and access to the club library (see rules elsewhere). Canadian regular and student memberships also have access to all the meetings, which otherwise are $5 per admission to the general public.

The membership fees are as follows:

Canadian Associate members $20.
U.S. Associate members $20 in U.S. funds.
Overseas Associate members $30 in U.S. funds.
Canadian Student members $20.
Canadian Regular members $30.

Membership applications are to be sent to:

TORONTO PET USERS GROUP
c/o Chris Bennett
381 Lawrence Avenue West
Toronto, Ontario, Canada
M5M 1B9

Include your phone number, no matter where you live and, if a student, your school affiliation.

TPUG SUSTAINING MEMBERS

Sustaining membership is open to those organizations who wish to support TPUG to a greater extent than through regular membership. The annual fee for sustaining members is $100.00. In return, these organizations are listed in the TORPET.

TPUG CLUB DISKS

To order TPUG club disks via the mail, just send $10 for each 4040/2031/1540/1541 disk and $12 for each 8050/8250 disk (payable in advance). This includes the price of the diskette, the labour involved to copy them and all postage and packaging charges. Do not send us any diskettes. The mailing address is:

TORONTO PET USERS GROUP
P.O. Box 100
Station "S" Toronto, Ontario
Canada M5M 4L6

Do not try to order any disk whose directory listing has not yet appeared in any issue of the TORPET. Most of the directory listings can be found in issue #12 (August/82) of the TORPET with cummulative updates printed in each new TORPET. Please **include your membership number and return address with all orders.**

Education Disks

The 50 Education Disks listed in issue No. 14 of The TORPET may be ordered in the normal way described above or the complete set may be obtained by sending $300 to:

Aurora Software
Att. Jennifer Godfrey
Box 1394
Haileybury, Ontario
Canada, P0J 1K0

The $300 includes the 50 diskettes, two hardcover binders, together with the documentation on all the programs. Orders must be prepaid, except in Canada where school boards may send a purchase order.

COPY TREE

The copy tree is a procedure whereby TPUG members can get a complete copy of the TPUG library in 4040 disk format.

Bonnar Beach
Horning's Mills, Ontario
Canada L0N 1J0
Phone 519/925-5376

Club Tapes

The procedure for ordering club tapes is to send $12.00 for each disk desired in tape format to:

RTC
10610 Bayview Plaza, Unit #18
Richmond Hill, Ontario
Canada L4C 3N8

Make all cheques or money orders payable to RTC and please **include your membership number and return address**

Most disks require two tapes and you will receive both tapes for the $12. For the few disks that will fit on a single tape an additional free tape will be sent.

RENT A MODEM

If you have never had the pleasure of signing on a BBS why not give Gord Campbell a call (492-9518) and try to rent one of the club modems for a month, you may find the BBS community to be a very high spirited active bunch of computerists, ranging from the beginner to the expert such as Jim Butterfield.

I say that I really think that Steve Punter has written a great Bulletin Board program, and compliment him on his efforts.

I cannot myself decide which is more of a challenge and pleasure, to be a user or to be a SYSOP. Why not try the BBS's out and experience the user side of it?

Trader's Floor at TPUG Conference

by David Williams

The TPUG Conference, to be held on May 14th and 15th, 1983, at George Brown College, Casa Loma Campus, in TORONTO will provide an opportunity for TPUG members to trade equipment, software etc.. Part of the conference area will be set aside for this purpose, and staff will be available to handle the sales, making it unnecessary for vendors to be present at all times.

All sellers and buyers will have to be TPUG members (this, of course, includes student and associate members). A fee of five dollars per item offered for sale will be charged to sellers, and will be payable whether or not the item is actually sold.

The prices of items will be fixed by their vendors, who will also be allowed to change the prices (for example to reduce the price of something which is failing to sell). However, an administrative charge of one dollar will be levied for every price change.

Vendors will also be allowed to choose whether to accept only cash in payment for an item, or whether personal cheques should also be accepted. If cheques are acceptable, the sales staff will take reasonable steps to verify the identy of purchasers, but TPUG will not accept liability in cases of cheques being dishonoured.

For security reasons, there will have to be limitations on the movement of equipment into and out of the sales area, but we will try to minimize the inconvenience which which these may cause.

Free Machine Language Presentation

TPUG has arranged with Jim Butterfield to have a one-day 'Introduction to Machine-language' presentation on May 14, in conjunction with the conference. The exact format will depend on the number of people who will be attending; therefore advance registration is REQUIRED. To register Gord Campbell a call at 492-9518 any evening up to February 20.

There will be no cost to TPUG members.

TPUG Sustaining Members
Questar International
Richvale Telecommunications
T. Eaton Co. Ltd.

CLASSIFIED ADS

5 Cents per word, with $1.00 Minimum.
Payment in advance to:
TORPET CLASSIFIED
P.O. Box 100
Station "S" Toronto, Ontario
Canada M5M 4L6

FOR SALE

An excellent Data Base called File Cabinet $35.00 to TPUG members only at that price. MFS-2 Interface $68.00 par version $82.00 serial version. Products reviewed in the Midnight. Progress Computer 7073 LynneTree Way Citrus Heights CA 95610 (6)

CATALOGS

PET/CBM ADD-ONS FREE CATALOG. ECX COMPUTER COMPANY has over 20 new ad-on circuits and software for your PET/CBM computer and peripherals. For a FREE CATALOG send a self addressed stamped envelope to: ECX COMPUTER COMPANY 2678 North Main St. Walnut Creek, California, 94596 (5)

-- FOR TRADE

Atari® Games On Your VIC-20?

The "CARDAPTER/1" will allow Video Game Cartridges designed for use on the Atari Video Computer System to be played on a standard VIC-20® $129.00

VIC-20 SOFTWARE

CD001 Froggee	$29.95
CD002 Centipod	$29.95
MD001 Snakman	$29.95
CM401 Paratrooper	$29.95
CS007 City Bomber & Minefield	$39.95
NU2003 Krazy Kong	$24.95
NU2006 Rescue From Nufon	$24.95
SYN002 Crabs	$18.95
SYN003 Cyclones*	$23.95
CS013 Astroblitz (K)	$59.95
CS017 Choplifter (K)	$59.95
UMI1604 Spiders of Mars (K)	$69.95
UMI1636 Cloud Burst (K)	$69.95

* Requires 3K Exp. (K) Cartridge

COMMODORE 64 SOFTWARE

Froggee (cassette)	$29.95
Pal 64 Assembler (disk)	$99.95
Coco (disk & cassette)	$59.95

BOOKS

Best of Transactor Vol. I	$12.00
Best of Transactor Vol. II	$16.00
Pet Graphics	$23.95
Library of Pet Subroutines	$23.95
PET Basic	$19.95
VIC20 Prog. Reference Guide	$29.95
Programming the PET/CBM	$39.95
Getting Acquainted with VIC20	

ADD POWER TO YOUR COMMODORE COMPUTER $119.95

Power is a programmer's utility package (in a 4k rom) that contains a series of new commands and utilities which are added to the screen editor and the basic interpreter. Power is a must for every serious CBM user.

MODEM from General Data Comm

Direct Connect — 300 BPS

$225.00

Quadra Rom **$44.95**

Allows four Roms to reside at one address

COMPUTER SUPPLIES

C10 Cassette Tapes	$1.00 ea
C20 Cassette Tapes	$1.25 ea
Cassette Boxes	.25 ea
Verbatim Disks M0525-01	$49.00/10
Memorex Disks 013481	$39.00/10
K10 Disk Storage Box	$5.95
Flip Sort Storage Box	$39.95
Printer Paper	
15 lb 9½x11 2850 sheets	$44.00
20 lb 9½x11 2200 sheets	$44.00

ORDERING INFORMATION

Cheque, Money Order, VISA, Master Card accepted. Ont. residents add 7% sales tax. Add 5% for shipping (minimum charge $1.00).

ELECTRONICS 2001 LTD.

5529 Yonge St., Willowdale, Ont. M2N 5S3
(416) 223-8400

VISA

MasterCard

Richvale Telecommunications

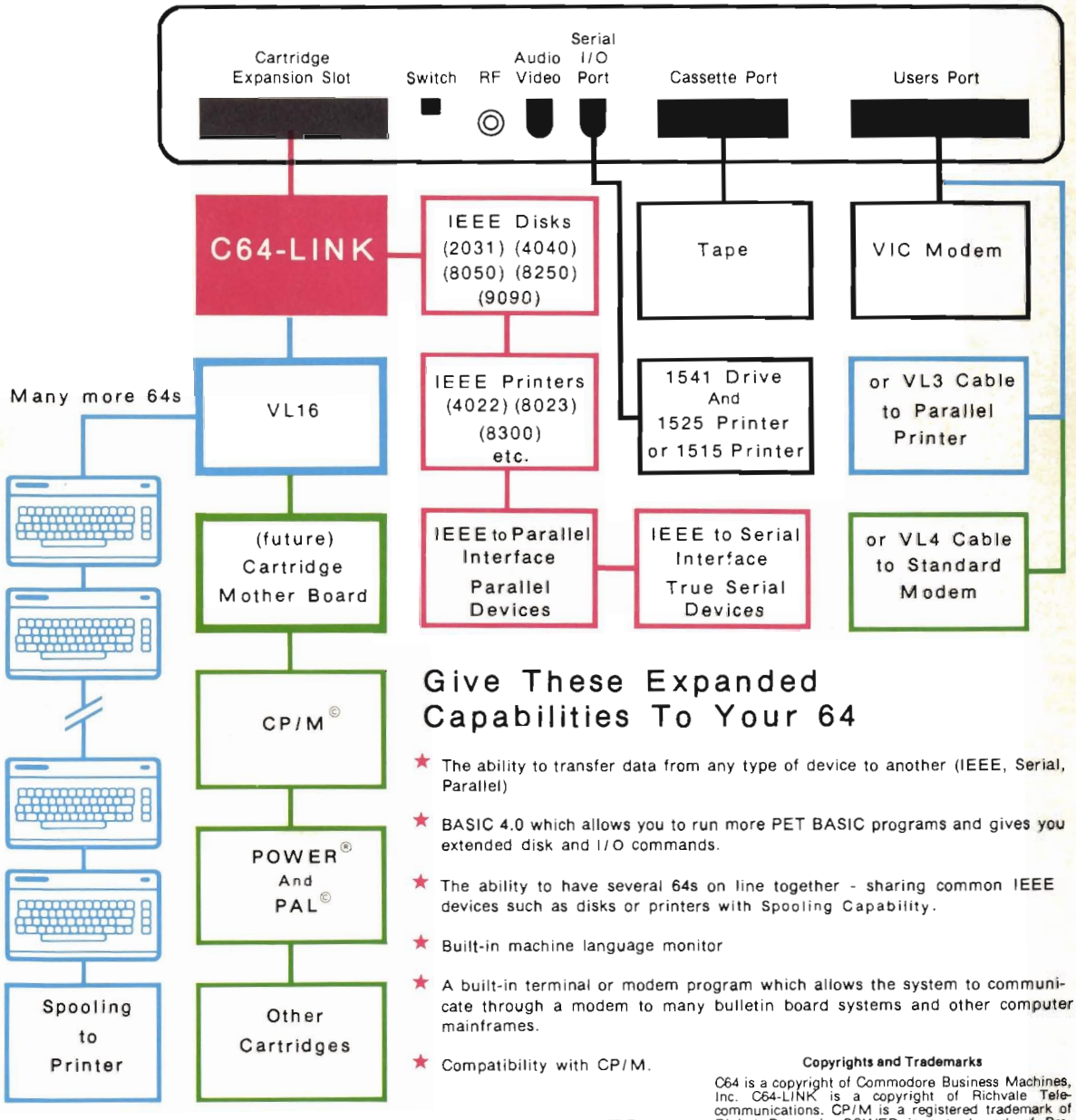
10610 BAYVIEW (Bayview Plaza)
RICHMOND HILL, ONTARIO, CANADA L4C 3N8
(416) 884-4165

$185 Canadian

C64-LINK©
The Smart 64

Also available for VIC 20

RTC RTC



Give These Expanded Capabilities To Your 64

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

Contact your local Commodore dealer or RTC.