COMMODORE WORLD

THE NEWS MAGAZINE FOR COMMODORE 64 & 128 USERS

Issue 16
August/September 1996
$4.95 U.S.

Commodore Demos Part Two
Max finds some Headroom on a Commodore

This is Your Brain on a Commodore
Simulate the way your brain recognizes patterns

In Review: Programming the 65816
The definitive book on 65816 programming

Plus: An Introduction to Programming the 65816

SECOND CLASS

Commodore World Software Blow-Out!

The purchasing department of CMD, in conjuction with Software Support International and Commodore World magazine, have come together to offer you four of the hottest game packages available today!

Whether you're passing for the game-winning touchdown, jumping through oncoming traffic, or dodging SAM missiles over North Vietnam, you are sure to enjoy these AAA rated games. Each of these packages represent some of the finest games within their class. They won't last long, so order yours now!

Flight

Sports

Arcade (Pack 1)

Qix - *Taito*
Frogger - *Sega*
Guerilla - *Data East*
Grand Prix Circuit - *Accolade*

Adv./Strategy (Pack 2)

Mean Streets - *Access*
Cut Throats - *Infocom*
Lords of Conquest - *E/A*
War in Middle Earth - *Melbourne*

Sports (Pack 3)

Monday Night Football - *Data East*
Jordan vs. Bird - *E/A*
Hardball - *Accolade*
Kings of the Beach - *E/A*

Air Combat (Pack 4)

High Roller- *Mindscape*
Sky Fox - *E/A*
Snow Strike - *Epyx*
Thud Ridge - *Three Sixty*

Only $20.00 per Package!

Plus Shipping & Handling

(Continental U.S. Add: 1-$5.00, 2-$5.50, 3-$6.00, $4-$6.50 All other destinations, see CMD main ad; Page 28)

To order call: 1-800-638-3263
(Mon.-Fri. 9 a.m. – 5 p.m. EST)

Adventure/Strategy

Arcade

With this special offer, buy three or more packages and get a 1670 (1200) baud modem FREE!

CONTENTS

ISSUE 16

AUGUST/SEPTEMBER 1996

COMMODORE WORLD

THE NEWS MAGAZINE FOR COMMODORE 64 & 128 USERS

GENERAL MANAGER
Charles R. Christianson

EDITOR
Doug Cotton

ADVERTISING SALES
Charles A. Christianson
(413) 525-0023

PHOTOGRAPHY
Wayne Wrubel

GRAPHIC ARTS
Doug Cotton

ELECTRONIC PRE-PRESS & PRINTING
Mansir/Holden

Cover Design by Doug Cotton

FEATURES

REVIEWS

COLUMNS

DEPARTMENTS

See Our NEW CHECKSUM Utility On Page 51

FROM THE EDITOR

A Mixed Bag...

There's no question that this most recent cycle has been a hectic one. Our new assistant editor didn't work out, so we're back to looking for someone to fill that position. In the meantime, all the editing, layout, and graphics creation that goes into each issue has fallen back into my lap. In addition, we've lost our photographer, Wayne, who was also CMD's repair technician. The latter spot has been filled, but we'll need to figure out what to do for photographs in upcoming issues. Despite these hurdles, we've managed to nudge the page count back up to 56, and I think we've produced a very well-rounded issue. Of course, I may be somewhat biased.

There's some bad news... a few hardware items have quietly dissappeared from the supply chain in the last few months. Notably missing in action: CMD's FD-4000, the Super 1750 Clone, SuperSnapshot, Action-Replay, VDC RAM Expansion kits, and disk drive RAM Expansion kits. These are sad losses, as there really aren't any products left that directly replace or duplicate what these products did. When suppliers run out of 1581 drives, no big deal, there are FD-2000 drives that can pretty much directly replace them; but losing both Super Snapshot and Action-Replay leaves users without any means to capture programs for archiving or operating from devices like CMD's HD. Expect to see some rise in the value of these items in the used market.

There are some positive things happening, as well, though... CMD's SuperCPU 64 has started shipping, and work on the 128 version is now under way. Maurice Randall has new versions of GeoFAX and GeoSHELL coming along soon, and he has also confirmed that he isn't far off from releasing a new GEOS driver/Configure package that will reportedly revolutionize device handling under GEOS. Nick Rossi's Novaterm 9.6 is now shipping on disk, and has brought us Zmodem, Ymodem-G, and a lot of other improvements. There are other software developments under way, too... Matt Desmond appears to be working regularly on his new version of Desterm, Electric Boys recently sent us some images created by a new C64 interlaced graphics editor that they are working on, and CMD is finishing up documentation for an English language version of GoDot. CMD is also heading up projects to produce a new 65816 assembler (SAS) and a machine language monitor (SuperJiffyMON), both targeted at SuperCPU developers; there are also indications that the long-awaited Menuette 128 may be back on track for release, along with an updated 64 version. So there are indeed some things for users to look forward to over the next few months.

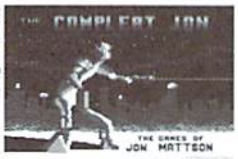
Meanwhile, there are some really nice things to see in this issue of Commodore World. Sherry Freedline brings us her second installment on Commodore demo programs; there's a very interesting article about how computers and human brains differ, along with a type-in program that lets the 64 emulate the brain's ability to recognize patterns; Brett Tabke introduces the 65816 to 6502/6510 programmers (we've added a useful opcode list and informative programming models to this); and Jayme Rice, President of TCUG, presents some excellent ideas on promoting growth of user groups. In our regular columns, find out what the User Port is good for (*Just For Starters*), how to create fractal images (*BASIC Instincts*), and how to use Internet File Transfer Protocol (*Carrier Detect*). We've also been rejoined this issue by Harold Stevens, Jr., who tackles the touchy subject of software piracy in *Over The Edge*. So what are doing still reading this editorial? Dive in!

Doug Cotton
Editor

LOADSTAR® MONTHLY

LOADSTAR is a monthly "magazine on disk" for the Commodore 64/128. Subscribers receive two 1541 disks (or one 1581 disk) in their mailbox every month packed with news, articles and programs. These non-PD, high-quality programs are written by the best home-based programmers in the field and edited by the crack LOADSTAR team of Fender Tucker and Jeff Jones. Subscription prices are at an all-time low of $69.95 for a 12-month subscription, or $19.95 for a three-month subscription. You may also elect to subscribe "by the month," where we charge your credit card $7.95 for each issue after it's shipped. We also offer the long line of standalone products below.

✔ NEW Games Disk! **The Compleat Jon:** 11 Games! The whole gamut of gaming is covered here: artificial intelligence, role-playing, mazes, fantasy, science fiction, education and even non-violence (which was a radical concept in its time). These eleven games are among the best ever published on LOADSTAR. Listed on the menu in chronological order, so you can see how Jon's style changed as the years rolled by. **1581 disk 0021D3 $20. 1541 disk #0038D5 $20**



✔ NEW Puzzle disk!
The Compleat Crossword: Every crossword puzzle published in Puzzle Page in one huge collection! 220 puzzles! It uses Barbara Schulak's CRUCIVERBALIST program to present the puzzles and allows you to "mark" a puzzle when it's solved so that you know which you've solved and which you haven't yet. Each 1541 disk contains 110 puzzles. **1581 Disk #0020D3 $20. Disk 1 (1541) #0036D5 $10.Disk 2 (1541) #0037D5 $10**

✔ NEW C-128 Productivity! **The Compleat Lee O:** Six of Lee O. Clinton's best serious programs for the C-128 80- column mode. Finance, auto expense, kitchen helper, genealogy, resume writing, mutual funds! **One 1541 disk #0032D5. One 1581 disk #0017D3 $10.00**

✔ NEW Word Search! **Super Star Search 1:** 200 original word search puzzles by Steven Thomas and Art Dudley, presented by John Serafino's modern point and click program. **One 1541 disk #0011D5 One 1581 disk #0008D3 $20.00**

✔ NEW Story disk! **The Compleat Prosequest '95:** NEW!!! A 1541 disk with all of the entries in the 1995 short story writing contest on it, including the three grand winners. **One 1541 disk #0035D5 One 1581 disk #0019D3 $5.00**

Soft Wear! **LOADSTAR T-Shirts:** Limited edition Fruit Of The Loom T-shirts. LOADSTAR's nemesis, Knees Calhoon, stands up to regular washing and drying. Where else can you find Commodore apparel? 50% Cotton//Polyester. **$15.00 each with** *free shipping*! **Small #960025, Medium #960125, Large #960225, X-Large #960325, XX-L #960425**

Tools! Tutorials! **The Compleat Programmer:** Best seller! Over Two megabytes of knowledge crammed and stuffed onto eight 5.25-inch disks or two 1581 disks! Plus tools, extensions, languages, assemblers, tutorials and utilities! **5.25 set #0005D5. 3.5-inch disk #0005D3. For $5 more, get C= Hacking MAG #0006D3 (on 3.5-inch disks only and NOT available separately)** to complete your programming set. **$20.00**

PS Graphics! **Compleat PS Vol. 1** (The Print Shop by Broderbund or Printmaster required). Over 1300 artistic and never before published PRINT SHOP images. Scan through the many PRINT SHOP images sequentially, by name, or by group number. Press a key and save the graphic you want in 2-block, 3-block and even PRINTMASTER graphic files! All that plus a printed guide! Each volume is $20.00. **Vol. 1: C-64/128 3.5-inch disk item #0001d3. 5.25-inch disks item #0009d5. Vol. 2 (graphics from past LS issues): C-64/128 3.5-inch disk item #0002d3. 5.25-inch disks item #0010d5.**

GEOS Clipart

LOADSTAR presents the biggest Geos collection of clip art and fonts ever offered at one time. All of the Geos art that's ever appeared on LOADSTAR, as well as some great files from Geos fanatic Dick Estel, are available on twenty 5.25 inch disks or eight 3.5 inch disks. Most of this has never been seen before! Use these graphics in your GeoPaint, GeoWrite and GeoPublish documents or convert to FGM with FGM utilities. Spiff up your GeoFAX documents with the appropriate graphic -- every time! Prices are $20 for any two 3.5 inch disks, or any five 5.25 inch disks. You can purchase the whole collection for $75 for either version. Call LOADSTAR toll-free at 1-800-594-3370 or 1-318-221-8718 to order by credit card. Or send check or money order and specify (by LG number) which disks you want.

5.25-INCH DISKS

Disk 01 - RAILS: Railroad art from Europe and the USA **#0012D5**

Disk 02 - VEHICLES/TAROT: Artwork of old and new autos; excellent geoPaint drawings of the Tarot card set **#0013D5**

Disk 03 - CLIP ART: Includes converted MacPaint files that have never before been available in Commodore format **#0014D5**

Disk 04 - OTTOWA/PRIME CLIPS: Artwork of the main landmarks of Ottowa; plus high quality public domain clip art **#0015D5**

Disk 05 - FONTS: More than 30 fonts from past issues of LOADSTAR, plus articles (in geoWrite format) on creating fonts. Also two ready-made headers for use with your own documents, one a picture of a mail truck; the other reading FROM THE DESK OF **#0016D5**

Disk 06, Disk 07, and **Disk 08** - geoPaint and Photo Album files with the great clip artwork featured on past LOADSTARS - Includes GeoCurmudgeon, Anamalia I and II, Australian Animals, Valentine art and many more **#0017D5, #0018D5, #0019D5**

Disk 09 - GOODYKOONTZ FILES - Jasper Goodykoontz, born in Indiana in 1855, produced Goodykoontz's Perpetual Calendar and General Reference Manual (A Book for the Millions). This disk includes scans from the book of a wide array of subjects -- Gestures and Attitudes, Poultry, Craniology, and more. **#0020D5**

Disk 10 - OLD WEST: Scanned Artwork from Dick Estel's FRD Software - mostly woodcut style art of the old west, gold rush days and pioneer scenes. **#0021D5**

Disk 11: J. Neely Art/Animals: Jennifer Neely works with a wide variety of subject matter and materials. Disk contains some of her favorites, scanned into geoPaint format. Side 2 is a collection of scanned artwork of animals from FRD Software **#0022D5**

Free Shipping! For Every item on this page!

Disk 12 - HOLIDAY: Artwork for New Years, Valentine's, St. Patrick's Day, Halloween, Thanksgiving and Christmas **#0023D5**

Disk 13 - PEOPLE/FACES: Scenes of people and faces from FRD Software **#0024D5**

Disk 14 - FRD CLASSICS: Dick's choice of the best of the FRD collection **#0025D5**

Disk 15 - DINOS/CLASSICS: Dinosaurs and other prehistoric beasts, as well as more first choice artwork from FRD. **#0026D5**

Disk 16 - SPORTS/MISC: Dozens of sports-related clips **#0027D5**

Disk 17 - OFFICE AND SCHOOL: Clips to be used at work and around the house **#0028D5**

Disk 18 -MUSIC & MORE SCHOOL CLIPS #0029D5

Disk 19 - SEASONAL AND HOLIDAYS: A clip for any occasion **#0030D5**

Disk 20 - SEASONAL AND HOLIDAYS: A clip for any occasion **#0031D5**

3.5 INCH DISKS

The 3.5" disks are roughly equivalent to two and a half 5.25" disks.

Disk 1: Equals disks 1, 2, 4B #0009D3
Disk 2: Equals disks 3, 6, 7A #0010D3
Disk 3: Equals disks 5, 8, 7B #0011D3
Disk 4: Equals 9, 10, 11A #0012D3
Disk 5: Equals 12, 13, 11B #0013D3
Disk 6: Equivalent of Disks 14, 15and some bonus files not on 5.25" disks **#0014D3**
Disk 7: Sports, Office and school, Music #015D3
Disk 8: Music, Holiday and Seasonal #016D3
For your convenience, GeoViewer is included on each volume. GEOS 2.0 is suggested.

Diskfulla Card Games! **The Compleat Maurice:** A compilation of 26 solitaire card games written by Maurice Jones, the acknowledged master of card game simulations for the C-64/128. There's even a brand new, never before published game called Boomerang. **Two 5.25 inch disks #0007D5 or one 3.5 inch disk #0007D3. $20.00 postage paid!**

Oodles Of Stunning Art! Tutorials! **The Compleat Walt:** During LOADSTAR's first ten years we have published 24 of Walt Harned's slideshows and multimedia events. Now we've gathered them into one huge collection: seven 5.25 inch disks or three 3.5 inch disks! There are over 250 pictures, including some that have never been published. The greatest one-man collection of art on any computer platform. **5.25-inch disks order #070425 3.5-inch disks order #070423. $20.00 postage paid!**

Loadstar Order form ☎ 1-800-594-3370

QTY	Description	Item #	Price ea.	Total

Use extra sheet of paper for large orders

Name ____________________

Address ____________________

City ____________________ State ________ Postal Code ________

Total Enclosed: ____________

❑ Check/money order made payable to "Loadstar" in US Funds
❑ MasterCard ❑ Visa ❑ American Express ❑ Discover
Card # ____________________
Exp Date _____/_____ Authorized signature: ____________________
Subscribers: I prefer ❑ 5.25-inch diskettes ❑ 3.5-inch diskettes

LOADSTAR
P.O. Box 30008, Shreveport, LA 71130-0008
Questions: 1-318-221-8718 Fax 1-318-221-8870

BACKTALK

LETTERS AND QUESTIONS FROM OUR READERS

Who's Your PAL?

After running our first installment of "Commodore Demos" (CW#15, pg. 20) a few of our readers were left with the nagging question, "What do NTSC and PAL mean?"

NTSC and PAL are video starndards used throughout the world. The NTSC video standard (named for the National Television System Committee) is used in most American countries, Japan, Taiwan, Korea, and others. Most of the rest of the countries where we commonly see Commodore computers in use (UK, Germany, Australia and others) have adopted the PAL (Phase Alternating-Line standards) system.

While the actual differnces between these two standards is rather technical, what you see on your monitor is the effect of several hundred lines of video information being written to it. Each of these lines is referred to as a "raster scan line". NTSC has 525 lines and is generally operated on a frequency of 60 Hz, while PAL has 625 lines and usually operates on a 50 Hz source.

Why is this important? Well, the average program on a Commodore could generally care less which video standard you're using for display purposes; however, specialized programming such as that commonly used in demos often uses video "raster" timing. The effect of operating such a program on a machine that doesn't use the intended video standard might be anything from graphics that are jittery to disk lockups caused by custom disk routines.

Perhaps you're wondering what it is that determines whether a specific Commodore computer is NTSC or PAL? There are three hardware differences: the clock crystal, the VIC chip, and a jumper located on the circuit board. If we get an opportunity, we'll try to present those differences and how to change the computer from one standard to the other in a future issue.

FD/1581 Format Tip

Maurice Randall, our geoProgrammist columnist, recently stumbled onto this 'gotcha' and a handy work-around tip for users that have both the CMD FD Series floppy disk drive, and a Commodore 1581. We'll let Maurice tell the story in his own words:

For all you FD Series owners, here's my 'Trick of the Month'...

First, the scenario...

Okay, you've got an FD drive and you've also got a 1581. You use both of them. You've got this 800K DD disk that you've been using in your FD, and it's been formatted with one native partition on it. It's been working just fine and then you decided to reformat it in your 1581. Now, you are using it in your 1581 as a 1581 disk, no more native partition. It works just fine there as well, and you decide to also use this disk in your FD once again—but the FD won't read it! But wait... The FD is *supposed* to be able to read 1581 disks, right? Not this one. Now, you've used disks formatted on the 1581 in the FD before with no problems at all, but this one just won't work in the FD anymore. But it still works fine in the 1581. Normally, you'd be more inclined to think that the 1581 would have a problem with an FD formatted disk, rather than the other way around.

If you've never seen this obscure problem, try it. Format a double-density disk in your FD with one native partition on it. Then reformat it in your 1581. Your FD will no longer accept it.

Now for the cure (and you can use either your 1581 or your FD to fix the disk). The problem stems from the fact that this disk still contains a system partition. Check it out. Put it in your FD and send an @$=P command to it. You will get a partition directory and it will still show a native partition on it. And that's why the FD can't read the disk—because it can't find a valid directory header on track one where CMD Native partitions normally have them. To fix this disk, you need to wipe out the system partition. You can do this very easily by sending a command to your drive. And yes, the 1581 can be instructed to wipe out the system partition without harming any other data that you might have on the drive. Just send the following command to the drive:

```
@"uØFbpjpbb"
```

Let me explain this closer. Some of those characters should come out as reverse characters on your screen. The capital 'F', each 'b', and the 'j' should be reverse characters. To get the reverse capital F, you press the F3 key. The reverse b is obtained by pressing CONTROL-b. And likewise, the reverse j is done with CONTROL-j. Be sure to press the '@' key followed by a quote and then the rest of the characters. End it with a quote and then press return. Your 1581 will promptly trash the system partition and the FD will once again be able to read this disk. If you perform this on your FD, it will also work, but to read it you will have to first remove the disk from the drive and then reinsert it. The FD will then be able to display the directory every time thereafter.

With JiffyDOS, the above string can be sent to the drive. Without JiffyDOS, or if you prefer, you can make a basic program and keep it with all your other little utilities. The following will do:

```
1Ø open15,8,15
2Ø print#15,"uØ"chr$(134)chr
$(2)+"p"+chr$(1Ø)+"p"+chr$(2)c
hr$(2)
3Ø close15
```

Then just save it to a disk as 'WIPESYS' or something like that. Anytime you need to trash the system partition on an 800K disk, just load it up and run it. In line 10, make sure that the '8' gets changed to whatever device number you have your 1581 or FD set as, depending on which drive contains the disk that needs the system partition wiped out on.

Hope someone can get some use out of this little tidbit.

- Maurice

SOFTWARE SUPPORT INTERNATIONAL

We Carry America's Largest Selection of C-64/C-128 Software!

ENTERTAINMENT

Arkanoid 2	$7.97	Pacman	$9.97
Beyond Dark Castle	$7.97	Plundered Hearts	$12.97
Beyond Zork 128	$12.97	Portal	$14.97
Demon Stalkers	$8.97	Powerdrift	$7.97
Double Dragon 2	$9.97	Questron 2	$9.97
Heavy Metal	$12.97	Rampage	$7.97
Heros of the Lance	$14.97	Rendevous With Rama	$7.97
Jeopardy 1-2-Jr	$9.97	Roadwars	$7.97
Keys to Maramon	$9.97	Sidearms	$7.97
Mean Streets	$9.97	Steel Thunder	$9.97
Ms Pacman	$9.97	Strip Poker	$14.97
Napoleon In Russia	$7.97	Trump Castle Casino	$12.97
Ogre	$9.97	Wheel Fortune 1-2-3	$9.97

SPORTS & FLIGHT

4th & Inches	$7.97	Tony LaRussa Baseball	$7.97
Champshp Baseball	$7.97	Tournament Tennis	$7.97
Champn Basketball	$7.97	WWF Wrestling	$7.97
Dave Winfield Batter Up	$9.97	Acrojet	$9.97
Fast Break	$7.97	Apache Strike	$7.97
Hardball	$7.97	Blue Angels	$7.97
Jack Nicklaus Golf	$9.97	F-14 Tomcat	$9.97
Jordan vs Bird	$7.97	F-19 Stealth Fighter	$12.97
Leaderboard Golf	$9.97	Flight Sim Games	$4.97
Pro Football Facts	$9.97	High Roller	$7.97
Pro Soccer	$7.97	Jet Combat Sim	$7.97
Pure Stat Baseball	$9.97	Skyfox	$7.97
Sporting News Baseball	$9.97	Super Huey I	$7.97
Star Rank Boxing	$7.97	Top Gunner	$9.97

ACCESSORIES

C-128 RGB Cable	$17.95	Floppy Disk Notcher	$6.95
C-64 Color Cable	$12.95	Dust Covers - specify	$7.95
Computer Hand 2	$6.95	Ergostick Joystick	$16.95
Convert A Com	$24.95	CBM 1200 Baud	$9.95
Disk Bank 10/3.5"	$2.95	Aprotek 2400 Baud	$49.95
Disk Bank 10/5.25"	$2.95	1351 Smart Mouse	$44.95
Disk Bank 100/3.5"	$12.95	Mouse Holder	$4.95
Disk Bank 100/5.25"	$12.95	MW 350 Interface	$44.95
Disk Bank 70/5.25"	$7.95	64 Power Supplies	$34.95
Disk Mailers	$0.39	Printer Ribbons	CALL
Drive Cleaners	$6.95	Userport Expander	$24.95
Serial Cable 4 or 6ft	$8.95	3.5" 10 cnt.Floppy	$7.95
Drive Power Cable	$7.95	5.25" 20 cnt.Floppy	$4.95
User Port Cable	$15.95	Diskette Sleeves (25)	$2.00
Com Modem Adapter	$15.95	Write Protect Tabs(100)	$1.00

PRODUCTIVITY

1750 Super Clone	$99.95	Geos 128 v2	$44.97
Basic Compiler 64	$12.97	GeoPublish	$34.97
Big Blue Reader	$29.97	Graphic Label Wizard	$14.97
B/W Prog Tools	$14.97	Graphics Basic	$9.97
B/W Power C	$9.97	Home Designer 128	$24.97
B/W Turbo Cartridge	$17.97	Manager, The	$12.97
C128 Graphics Bundle	$29.97	Maverick v5	$24.95
Christmas Model Kit	$9.97	Model Diet	$9.97
CSM Protection Man 1	$14.97	Newsroom	$14.97
CSM Protection Man 2	$19.97	On Line Help	$9.97
Data Manager 2	$14.97	Postcards	$14.97
Designer's Pencil	$9.97	Printmaster Plus	$19.97
Drive Align 1541/71	$12.97	Superbase 64 or 128	$19.97
Easy Working Tri Pak	$9.97	Swiftcalc w/Sideways	$14.97
Geos 64 v2	$39.97	Word Writer 5	$19.95

EDUCATIONAL

Early Learning Friends	$9.97	Snoopy Sky Scramble	$9.97
European Nations & Loc	$9.97	Stickybear Math	$9.97
Expeditions	$9.97	Typing Tutor 4	$9.97
Keys to Typing	$9.97	WizType/Wiz Math	$7.97
Memory Manor Cart	$9.97	Word Attack	$9.97
Magic Spells	$9.97	Word Spinner	$9.97

REFURBISHED HARDWARE

C-128 Keyboard w/PS	$119.95	80 Col Monitor	$139.95
1541 Disk Drive	$49.95		
1571 Disk Drive	$89.95		
1541 Clone Drive	$39.95		
40 Col Monitor	$99.95		

MPS-803 Printer With Tractor Feed & Brand New Ribbon Only $49.95

Items Listed Above Do Not Include Shipping. U.S 48 States - Add $5.50 per order. Alaska, Hawaii & Canada - add $5.50 for the first piece and $1.00 per each additional piece per shipment. Second Day Air shipping is available. Call for shipping charges. Call Or Write For Your Free c-64/128 Catalog Listing HUNDREDS Of Commodore Products And Special Offers For Your Computer. Our Order Takers Are On Duty 6:00 a.m. - 5:00 p.m. M - F and 7:00 a.m. - 3:00 p.m. Sat. - Pacific Time.

SOFTWARE SUPPORT

Software Support Int.
2700 N.E. Andresen Rd.
Suite D-4
Vancouver, Wa 98661
(360) 695-1393
E-Mail To: software@pacifier.com

CALL TOLL FREE TODAY! 1-800-356-1179

Major Credit Cards Accepted.

Commodore Trivia

by Jim Brain

Welcome to another edition of Commodore Trivia. As many of you may know, these trivia questions and answers have been donated by me to the Commodore community at large. Unlike other articles in *Commodore World*, these trivia questions have been placed in the public domain. I ask only that the trivia questions remain intact and unchanged, and that my name and address appear somewhere so users can contact me. The trivia is also used for a contest I run on the Internet; contact me at the included address for more information. Because curiosity has the best of me, I always welcome a note or postcard detailing where the trivia goes. I also welcome new questions—provided they come with the answers. Enjoy!

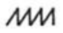
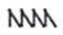
Jim Brain
Brain Innovations, Inc.
10710 Bruhn Ave
Bennington, NE 68007
j.brain@ieee.org

COMMODORE TRIVIA #15 QUESTIONS

$0E0 What is the difference(s) between the Newtronics 1541 and the 1541C? (only one difference is needed)

$0E1 What happens when you type 35072121 in direct mode on the C64 and hit return?

$0E2 If a SID chip is producing a "sawtooth waveform", does the waveform look like:

a)  or
b)  ?

$0E3 On BASIC 2.0, what special precaution(s) must one take when working with relative files? (only one is needed)

$0E4 What incompatibility existed between C128 Rev. 0 ROMS and the REU?

$0E5 What can trigger an NMI interrupt? (count all sources on one chip as one)

$0E6 What can trigger an IRQ interrupt? (count all sources on one chip as one)

$0E7 Where is the ROM in a 1541 located in the 64K memory map?

$0E8 Which VIA on the 1541 is hooked to the read/write head?

$0E9 In the Commodore DOS, what bit in the file type byte denotes a "locked" file?

$0EA If files are "locked" under Commodore DOS, under what condition(s) may the file be changed?

$0EB How big can a program file be on a 1541 or similar?

$0EC Under BASIC 2.0, how does one open a random access file on a disk drive?

$0ED A file that has a '*' immediately before the filetype is called a __________ file.

$0EE We know the 1541 and similar drives have 5 internal buffer areas, but how many does an 8050 drive have?

$0EF On a "save-with-replace", where is the location of the first track and sector of the new copy of the program saved in the directory entry for the old copy?

COMMODORE TRIVIA #14 ANSWERS

$0D0 The original PET had 73 calculator-style keys that were laid out in a rectangular matrix, not typewriter-style.

$0D1 SYS 32800,123,45,6. The screen will clear, and the software and hardware developers on the 128 project will be named. The exact text is as follows:

[RVS] Brought to you by...
Software:
Fred Bowen
Terry Ryan
Von Ertwine

Hardware:
Bil Herd
Dave Haynie
Frank Palaia

[RVS]Link arms, don't make them.

$0D2 The "original" PET came in two configurations, 4K and 8K, so:

The PET 2001-4 had 3071 bytes.
The PET 2001-8 had 7167 bytes.

$0D3 Sys 64790

$0D4 I know some of these are correct, but the sheer size of the list prevents me from checking them ALL out.

FAT 40XX series
80XX series
PC-10 (I suspect a number of IBM clones did, and these things have no consistent naming convention across country boundaries.)
PC-20
Amiga 1000
SP9000 (SuperPET)

$0D5 31743 bytes.

$0D6 a) 3 sockets.
b) 2 sockets.
c) 1 socket.
d) 1 socket.

$0D7 The German version had additional keybaord drivers for umlaut characters and dead keys.

$0D8 None other than the infamous Jim Butterfield.

$0D9 PRINT is faster, since the sys approach must process the pokes before the sys, which are very slow.

$0DA Inside the top case of the Amiga (1000). There is an interesting footnote to this question. It seems that at least some original Amiga machines were labeled as Amiga (with no number). Then, at some later point, the number was added. In addition, Commodore produced some Amiga 1000 machines without the signatures, but most had the telltale handwriting on the inside of the case.

$0DB Assume carry was clear. If so, then $11 is the correct answer.

$0DC Its first number was 6567, and that is the number most people know it by, but Commodore produced a VIC-II using a new manufacturing process that was numbered the 8562.

$0DD Same here. The part number 6569 is the most remembered number, but an 8565 will work as well.

$0DE Note, for the purpose of the calculation I am performing, "pixels" refers to picture elements that can be adddress and modified using normal VIC modes, so there are 320*200 "pixels" on both the PAL and NTSC screens. (I probably should have stated this, but it is too late now.) Also, the screen refresh rates used in the calculations are those defined by the respective television standards (60Hz U.S., 50Hz European), even though the actual frequencies are off by a small percentage. (for example, the actual 50Hz refresh rate on European VIC-II chips was calculates as 50.124567Hz by Andreas Boose). So, the PAL draws 320*200*50 pixels per second = 3200000 pixels/s. NTSC draws 320*200*60 pixels per second = 3840000 pixles/s . Now, some people thought I meant the whole screen, not just the display area provided by the VIC-II chip. Well, I am not sure exactly how you calculate pixels on a screen, since the numbers could vary from display to display, but if we measure in scanlines:

PAL = 312 scanlines * 50 = 15600 scanliness
NTSC = 262 scanlines * 60 = 15720 scanliness

The NTSC machines wins both ways.

$0DF B is the correct answer, and there are a couple of reasons why:

1) 2- -2 takes longer to parse in the BASIC interpreter.
2) Commodore BASIC subtracts by complementing the sign of the second number and adding. This incurs extra time. There are even more subtle ones, but I leave them as an exercise for the reader. Send me your reasons why.

CMD Service Center

AFFORDABLE • FAST • DEPENDABLE

Call Today 1-800-638-3263

Our Team of Technicians are Among the Most Qualified in the Industry! Why Settle for Anything Less Than the Best?

We repair the following equipment: Commodore C-64, 64C, SX-64, C-128 and C128-D computers; 1541, 1541C, 1571 and 1581 Disk Drives plus CMD Devices. JiffyDOS Installations a specialty. All repairs warranted for 30 days. Minimum charge $35.00 plus parts and return shipping. You must contact CMD for authorization before sending any equipment.
Creative Micro Designs, Inc. P.O. Box 646 E. Longmeadow, MA 01028

Computer Bargain Store

(801)466-8084

Specializing in NEW and USED Commodore Hardware, Software and Accessories at excellent prices.

Send $1.00 for a HUGE list of products. Office Hours: 11:30 - 6:30 MST. Visa, MC, Discover and American Express accepted.

3366 South 2300 East, Salt Lake City, UT 84109

ON THE HORIZON

Commodore and Computer Industry News

Genie Sold Again, Launches Genie Interactive on the Internet
Just as things were beginning to settle back down with Genie (see "CMD Captures Genie", CW#15, pg. 9), the announcement came that the service has once again changed hands.

True to the indications we reported last issue, Yovelle Reniassance had smoothed over relations with General Electric Information Services, who sold the service earlier this year. But only a few short weeks later, it was announced that the service had now been sold to IDT, a US-based Internet and long distance service provider. IDT has already begun testing of an Internet-based version of Genie, called Genie Interactive (http://www.genie.com). The new site will place a heavy emphasis on integrating the wide range of multimedia options available to graphical browser users, but is also being structured to allow text browser access. IDT hopes to utilize the new service to draw users who already have Internet access into the Genie fold. Despite the emphasis on the Internet venture, the company has also assured Genie sysops that the classic Genie online service remains viable, and will continue to operate as it has in the past.

Loadstar Letter Goes Subscription, Merges with Underground

According to Loadstar's "Grand Exalted Mojo" Fender Tucker, the Loadstar Letter #36 shipped with Loadstar #146 will be the last free issue provided to Loadstarites.

Tucker notes, however, that this doesn't spell and end for the Commodore newsletter, but rather a new beginning. The all-new subscription-based Loadstar letter was slated to begin in August, with an annual subscription rate of $12.00 per year for charter subscribers. Subscriptions are available by sending a check to LOADSTAR Letter, P.O. Box 30008, Shreveport LA 71130.

Loadstar cites rising costs as the reason for the change in policy, adding that their LOADSTAR disk publication will remain at its all-time low of $69.95 per year.

On a related note, Scott Eggleston recently announced that he would merge his Underground publication with the Loadstar Letter. The following notice was to be sent to Underground subscribers:

Dear Underground Subscriber,

I must first thank you for having faith in our little publication that started over two years ago. I had a desire to put out a quality publication using Commodore equipment, and for the most part, I think I have succeeded.

I realize it is always hard to send money to an indpendent magazine, as so many have dried up (along with all funds) in the past. I was determined that this would never happen to the Underground, that I would never leave my customers high and dry, with nothing to show for their hard-earned dollars.

There have been some real changes in my life recently, changes that have made me reevalutate my most precious commodity: time. With a new job, family demands, and filmmaking dreams, my time for writing and publishing has become limited.

As a result, the Underground will be merging with the Loadstar Letter, effective immediately. All remaining subscriptions will be fulfilled by the LL, and I will be brought aboard as an Associate Editor.

In my opinion, this is a win-win situation for everyone. Everybody continues to get a good publication for their money. I get to keep writing about one of my favorite topics, and have time for my other pursuits. The Loadstar Letter gets more subscribers. I feel this is a good move for everyone involved—nobody loses.

If you have never heard of this monthly publication, let me bring you up to date. Originally the LL was included as a freebie with Loadstar's fine disk publication. Strictly a liability, it was deemed that as a subscribed-to newsletter, it could pay for itself. It will be twelve full-size pages, which may be down from the Underground's twenty, but it will arrive every month, something the Underground could never do. Trust me, you'll be getting a lot for your money with the Loadstar Letter.

The Head Editor Honcho Guy is Jeff Jones, a great writer and equally great programmer. I look forward to having my work next to Jeff's, and am honored to have been considered for this position. With both of us writing together, you're going to end up with an informative and formidable source for Commodore info.

You should expect to see your first issue of the Loadstar Letter about the same time as Underground #15 would have shown up. Although, to be honest, it'll probably show up sooner, as I've been a bit tardy with my issues as of late.

There are also a couple of other topics I need to address. Back Issues of the Underground will still be available from me at $2.50 per issue. The Underware disk will obviously no longer be available, but Tom Adams [president of Meeting Commodore Users Through the Mail] has informed me that he will copy back issues of the disk for as long as people will want them, or as long as he is able to do so.

In closing, let me just say that I have enjoyed producing the Underground for you, and was always thrilled to read your letters telling me of some useful information that you gleaned from it. I still have a lot of good articles in my head, and you'll get to read them all (along with Jeff's), in upcoming issues of the Loadstar Letter.

If for any reason you need to contact me, please do so. Thanks again for your support... see you outside the mainstream!

Sincerely,
Scott Eggleston
former Editor, the Underground
Associate Editor, The Loadstar Letter

New Commodore Discovery Debuts

So, what's new in the world of Commodore computing? That's at least a part of the focus of *disC=overy*, a new online publication from Mike Gordillo, Steven Judd, and Ernest Stokes. From a perusal of the first issue of this publication, the apparent spin is to document both old and new technical issues and techniques as they apply to the Commodore—though the editors have mixed in a couple of articles which could be considered as "lighter" reading. For the most part, though, this debut issue appears to be very much like *Commodore Hacking*, for those of you who are familiar with that long-standing bastion of techno-enthusiasm.

Included in this first issue are articles on IFLI, TRI-FLI (a new FLI graphics scheme under development), the JCH sound editor, VDC timing, upgrading the VDC RAM to 64K, C128 CP/M, and more.

To get the current issue, you'll need Internet Email or World Wide Web access. On the Web, point your browser to *http://www.eskimo.com/~dray/discovery.html*. If you don't have Web access, send an Email request to the Editor-in-Chief at *s0621126@dominic.barry.edu*.

Threshold Products Sells Out

September 20th, 1996—Threshold Productions International is glad to announce that the takeover process of our operations by Arkanix Labs is finally complete. Arkanix Labs is taking over all day-to-day operations of TPI and will fulfill any and all of TPI's prior obligations. We feel Arkanix Labs will be able to better realize the ideals previously set by TPI. Our hopes are that Araknix Labs can provide for the customer where we have lacked in the past therefore insuring a bright future for the Commodore 64/128 community.

Arkanix Labs will be using the Seattle address for all mailings. The previous message number for TPI is now defunct.

Jonathan Mines [formerly of TPI] will continue to provide the Driven mail subscriptions. For further information contact Petar Strinic (*petars@arkanixlabs.com*). We have a WWW homepage at *http://www.arkanixlabs.com.*

[A] Letter from Jonathan Mines follows...

A dark day for the NTSC Commodore community? No, but a bright future isborn. Were I have failed in the past with TPI I believe Arkanix Labs will do a far superior job.

First off, anybody with orders will get their packages. There have been delays, but I really didn't forsee this takeover causing this much trouble. I'm sorry for the delays, I will be handling all packages myself in the next week.

What happened? Why is TPI gone? I simply couldn't keep my head above water much longer the way things were going. TPI just grew up too fast, I couldn't handle the load and needed to offload some work. I tried it, but the person I hired failed to help. Then an old friend came along, someone I've worked alongside before and this deal happened.

Will I be a part of Arkanix Labs? Right now I'm helping setup everything the right way, the way it should have been done with TPI. We have a staff now, so things will get done on a proper time schedule. My main job with Arkanix Labs will be handling all the financial work, something I do daily with my "real job". Besides that I'll be handling the legal end of the company.

But the best thing about having everything off my shoulders, now I can get some programming done! For the last two years I've put project after project off because of no time. Finally I get back to what I do best on this C64!

Anyway, I have explained everything, I think. Any questions about anything can be sent to me at the email address which follows. Feel free to email anything, I like to chat!

Jon Mines
jon.mines@arkanixlabs.com
tpinfo@eskimo.com

Performance Peripherals, Inc. (PPI) Bows Out

CW writer Gaelyne Gasson recently reported that Performance Peripherals, Inc., will no longer be servicing the Commodore market. Gasson confirmed this in a phone conversation with PPI founder Peter Fiset, who stated that the company is moving its focus to other projects. According to Gasson, PPI will no longer provide any support or repairs for their Commodore products, and has ceased all manufacturing of these products. The company will continue to sell products until present inventories are depleted.

SuperCPU Update

Initial reports from users on the compatibility, speed and usability of the SuperCPU 64 have been positive. But like any new product of this level of complexity, the first production run did find a few machines—specifically Commodore 64's utilizing the Revision E board—that were too far out of specification to allow for reliable operation. This number was extremely small, though, accounting for only six units, with the first of these reported by a user in Germany. CMD immediately investigated the situation, and halted production until modifications to resolve the problems could be developed and implemented.

What was discovered was that a small percentage of Commodore 64's have incorrect relationship between the Dot Clock and the Phase 2 Clock. The additional loading factor presented by adding the SuperCPU to these machines, often already outfitted with REU's and/or RAMLink devices, is enough to create erratic operation with some of these machines. CMD was able to resolve this problem by re-generating a proper Phase 2 Clock locally on the SuperCPU. This modification has been incorporated into all new production units, and no further reports of these problems have been received since. SuperCPU users experiencing erratic operation should contact CMD to arrange having their unit upgraded.

Meanwhile, CMD is continuing to work on the 128 version, the RAM expansion card, the developer's package, and several other related projects. However, no firm dates have been set at this time for release of these items.

ERROR CORRECTIONS

Issue 15, page 44, correct these lines for THE CHAOS GAME:

```
130 FORI=1024TO2023:POKEI,16:NEXT:REM SE
T COLOR (FOREGROUND*16+BACKGROUND)
155 Y=199:FOR X=0 TO 319:GOSUB 250:NEXT
250 B=8192+INT(Y/8)*320+8*INT(X/8)+(Y AN
D 7):POKEB,PEEK(B)OR M(X AND 7):RETURN
```

Issue 15, page 45, correct these lines for FERN CHAOS:

```
155 X1=.5*W:X2=.57*W:X3=.408*W:X4=.1075*
W
160 Y1=0*W:Y2=-.027*W:Y3=.0669*W:Y4=.2*W
250 B=8192+INT(TY/8)*320+8*INT(TX/8)+(TY
 AND 7):POKEB,PEEK(B)OR M(TX AND 7)
260 PRINT"{CLR/HOME}":POKE53265,27:POKE5
3272,21
```

Just For Starters

by Jason Compton

USEFUL ITEMS FOR THE USER PORT

Lately, we've tackled an introduction to input devices and cartridge port expansion products. Now it's time to turn our focus to the cartridge port's neighbor down the road, the user port. It's not the high-profile slot, to be sure. Nobody ever put a game on the user port, all of the accelerators live in the cartridge port, and even the multi-port expanders aren't as clever for the user port as they are for the cartridge port. Nevertheless, you can do a lot of clever things with a C-64 user port.

For a time, the user port was arguably the best way to expand a C-64 or 128. In recent years, the superior R&D has been done on cartridge port peripherals, but there is still a wealth of good use you can get out of your user port.

Modems

Perhaps the most obvious, and certainly one of the earliest, user port expansions, there are a number of modems out there which will plug directly into the C-64's user port. Commodore, and a number of other companies produced 300, 1200, and 2400 baud modems to get 64 users into the world of BBS's and online services. By today's standards, these modems will be slow and outdated, but in a pinch they may come in handy.

RS-232 (Serial) Interfaces

As time progressed in the computer industry, a number of products, including modems and some printers, strongly adopted the PC industry standard RS-232 serial port, found on PC clones and Amigas. It began to make much more sense for companies to build only one RS-232 version of their product, rather than catering to the 64's specific user port and serial bus. Not to be shut out, 64 developers found a way to add an RS-232 interface to the user port. Available both commercially (the Aprotek COM-MODEM and the Omnitronix Deluxe RS-232 Interface, among others) and as a project outlined in Commodore World #4, RS-232 interfaces open up a variety of new hardware for the 64.

Keep in mind, however, that serial printers are not as common as parallel printers (we'll get to that in just a bit) and the user port—RS-232 interfaces are generally limited to 2400 baud communication. Still, by the time 2400 baud modems were affordable there were hardly any companies willing to build them specifically for the 64 (Aprotek is the only one that comes to mind). With 2400 baud modems nearly worthless in most people's eyes and the RS-232 kit clocking in well under $25, you could turn a 64 into an exceptionally low-cost text terminal for about $30 if you shop well.

You'll also get the ability to use null modems, which were never constructed for the 64 user port specifically. Null modems are devices that allow you to hook up two machines directly to one another via the serial port and transfer files and data just as one would if calling a BBS with a "real" modem. Without getting into more expensive adapters that allow you to hook up a Commodore drive to a different computer, this is often the cheapest way to get data from a 64 into another type of machine. If you're looking for higher rates of speed, you've put yourself in SwiftLink territory and will need the cartridge-based expansion.

In theory, a standard external MIDI interface device could be plugged into such an RS-232 port, but I have yet to see any software that would support a serial MIDI box.

Centronics Parallel (Printer) Interfaces

Early in the game, printer manufacturers were more than happy to build 64-specific models of their products. But not many of these were particularly good—many suffered from not having true "descenders", or letters such as "g" which would drop below other letters. But when users craved for more, they would often find that there was simply not a 64-specific printer for their needs. Are you surprised that someone, in fact several someones, found a solution?

A great number of C-64 parallel interfaces have emerged on the market. The parallel port standard was established on the PC and is more or less fully supported by Amiga computers. (Macs, of course, have their own unique serial bus and no parallel bus.) For a time, the most popular of the parallel interfaces was the Xetec Super Graphix and Super Graphix Jr. These devices put the parallel printers on the 64's serial (disk) bus, and they also had to draw power from the 64's cassette port—not convenient if you, for example, have a dongle currently residing there, or are using a Datasette as a digitizer.

Enter the CMD geoCable II. It truly keeps the "parallel" in "parallel bus" and takes the printers away from the contested disk bus. The device is remarkably straightforward. It consists exactly of a circuit board, connectors at each end—one female user port connector to plug into the 64 and another male user port connector to plug other user port devices into for pass-through purposes, a 25-pin parallel port connector that faces sideways, and a large switch. That's all.

What it does, simply and elegantly, is turn your user port into a parallel port, allowing you to plug in the standard $3 cable that parallel printers use and be up and running in no time, with or without the specific GEOS drivers included. The switch disables the parallel port and enables the other user-port device currently plugged into the pass-through connector.

EPROM Burning

For hardware hackers, EPROMs are wonderful tools. A number of EPROM burners are available for the 64, but many of them are cartridge port devices. Datel manufactured the EPROMMER 64 to fit in the user port so as to avoid conflicts with other devices you might want to keep resident in your cartridge port. Not for the faint of heart, but if you know your way around programmable chips, here's your device.

Voice Synthesis

While some were mightily impressed with the SAM speech synthesis available for the 64 that used only the built-in SID chip, others wanted to do more. The Jameco Voice Synthesizer was actually an external box that connected to the 64 via custom cable, with included software to drive the unit. Just another example of what you can do with a 64 if you put your mind and a little bit of effort into it.

Currah Technologies built the Voice Messenger, a device that just keeps popping up. Its principle is roughly the same—it generates speech from internal circuitry and speakers. The heart of the unit is the EZ Speech software.

Video Digitization

Now we're talking. The big talk in the industry is multimedia, and all the Microsoft ads like to show you how much fun and productivity you can have when you put video on your computer. 64 peripherals did it a long time ago. Devices known as Computereyes and Video Byte allow you to "framegrab", or digitize single video images from a standard source. While they can't go so far as to offer you the sort of accuracy available on more expensive equipment, such as capturing a single frame in real-time, they can "slow-scan" images, such as an unmoving object or person in front of a camera (or a rock-solid pause, as found on laser discs and very, very good VCRs). Since these were built in the day before widespread support of some of the more clever 64 graphics modes was available, you'll be looking at greyscale or colorized output given the 64's standard 16-color set restrictions. But for greyscale images in particular, a good grab is plenty good to get the point across.

These devices will save their pictures into standard 64 file formats such as Koala. A real coup would be a new driver—perhaps a GoDot module—that could harness the greater color depth available to clever programmers and exploit that from a scanner such as these, or other devices.

Surely, this is not the final word on user port expansion, but it is a broad sampling of the sorts of clever expansions that are floating out there for the 64. Keep your eyes open—you never know when you'll have an opportunity to own your very own video digitizing station.

Paxtron CORPORATION

COMMODORE CHIPS AND PARTS

We also carry Amiga parts, Motherboards, Power Supplies and all surface mounted ICs

Paxtron CORPORATION

Upgrade Chips

6510, 6526, 6567,6569, 6581, 6522, 8502, 8562, 8500, 8563, 8564, 8721, 8722, 325302, 390059, 251968	**$8.95**
901225, 901226, 901227, 906114PLA	**$6.95**
251715, 251913, 8580,WD1772	**$10.95**
314972-03 (C128 ROM Upgrade)	**$24.95**

Motherboards

1541 (Refurb.)	**$14.95**
1541II (NEW)	**$19.00**
1571 (NEW)	**$19.50**
C-64 (NEW)	**$34.95**
C-64 (untested, as is, all chips)	**$17.95**
C-128 (NEW)	**$49.95**
C-128-D (NEW)	**$52.50**

Diagnostics & Manuals

Commodore Diagnostician: Guide to diagnose & fix C64/128/1541	**$6.95**
C128 Diagnostic Cartridge and manual	**$24.75**
Manuals: MPS 802 & 1230, 1084, 1902, 1581, 1526, 1571, 801, C64, C128	**Call**
We are now receiving manuals for all Commodore products from SMG	**Call**

Floppy Disk Drives

1541 (Refurb.)	**$44.50**
1541II w/ P.S. (NEW)	**$69.95**
1571 (Refurb.)	**$89.50**

Miscellaneous

C-64 Keyboard	**$14.95**
1351 Commodore Mouse	**$19.95**
Monitor Cables	**Call**
Used Monitors (Low Prices)	**Call**
1541/1571 Alignment Kit	**$15.95**
1230 CBM Printer	**$59.95**
1084S Motherboard w/ Flyback	**$79.95**
1084S Power Supply Board (Refurb.)	**$29.95**
C-64 Cabinet (Top/Bottom)	**$12.95**
Commodore Joystick	**$4.95**
256 x 1 (41256)	**$3.39**
1541 Serial Cable	**$4.95**
Video Cable	**$5.95**
Verbatim 5.25 SS/DD Disks(10 Pak)	**$1.99**
Amiga A500 (Rev.3) NEW Motherboard with all Chips	**$59.95**

Power Supplies

(All Power Supplies are NEW)

C-64 non-repairable	**$14.95**
C-64 repairable	**$19.95**
C-64 Heavy Duty (5.2 amps)	**$39.95**
C-128 Heavy Duty (5.2 Amps)	**$39.95**
5.2 amps for Ram Expander Units	**$39.95**
1541II external 110 volts	**$7.95**
1581 external 110 volts	**$7.95**
1084S Flyback (Phillips)	**$35.50**
1084-D1 Flyback	**$35.95**
1084 Flyback (Daewoo)	**$35.95**
Str54041 for 1802C Monitor	**$12.95**

- ***A500 Computer:*** Includes Power Supply and latest chips (eg: Agnus, 2.04 O/S). Plus choice of following: Starter Kit, Discovery Kit (incl. Kind Words, Deluxe Paint II) or Deluxe Kit **$149.95**
- ***CD32 Motherboard:*** Contains all chips(Lisa, Alice, Paula, 68020-16, ADV101) except the 512K memories .... **$79.95** With all memory **$104.50**
- ***Commodore C65 Motherboard:*** Contains all RAM operating at 3.5 MHz speed. Includes all chips (tested) **$69.95**
- ***1084S Motherboard with Flyback Transformer:*** This new CBM board will cure 90% of 1084S problems. Simply switch the PCB and your monitor problems are solved! This motherboard with factory mounted flyback, is the exact replacement and works with all 1084S monitors. It is also very easy to install **$79.95**
- ***Commodore 1976 Scientific Calculator:*** Save a piece of the past. Brand new with charger and manual **$8.95**
- ***C64 Emergency Diagnostic/Repair Kit:*** Each kit contains: 6526 CIA, fuse, 6581 SID, 6510 CPU, 906114 PLA, 6567 VIC, 901225 BASIC ROM, 901226 Character ROM and Commodore Diagnostician ($70 Value) **$49.95**
- ***C128/128-D Emergency Diagnostic/Repair Kit:*** Each kit contains: 6526 CIA, 8563/8564 80/40 column CRT Controllers, 390059-01 Character ROM, 8721 PLA, 8722 MMU, 6581 SID and Commodore Diagnostician ($70 Value) **$49.95**
- ***C64/128 Utility Cartridge:*** 30 programming functions on one plug-in cartridge **$8.95**

Paxtron CORPORATION 28 Grove Street Spring Valley, NY 10977 **ORDERS 800-595-5534 • 800-815-3241 • 888-PAXTRON** **Info 914-578-6522 • FAX 914-624-3239** Hours: 9AM - 5 PM EST Add $6.00 for UPS Charges We gladly accept:

Graphic Interpretation

by Paul Sullivan

GEOWRITE DONE RIGHT

While rereading the user's manual of Creative Micro Designs' "Perfect Print LQ for GEOS," a statement in the introduction stood out to me: "geoWrite is the most used application in GEOS." This holds quite true for myself—I used it daily (and nightly) through college, and still use it to create articles, calligraphy, resumes and newsletters. Over the years numerous programmers have invented different applications, drivers and desk accessories that can boost geoWrite's capabilities, and some of these may have never received much publicity following QuantumLink's demise in 1994. Thus it seemed appropriate to make this the focus of this issue's geoSphere article. In this two-part series, we will look at some tools and techniques that can help make this great application work even better for you!

For this series' first part, we will look at improving input: text, fonts, directories and formatting. In the second part, printer output and quality will be discussed.

Let's begin at the beginning, before even entering geoWrite. Do you have a mouse configured as your input driver? And if so, has the uselessness of the right button ever made you think, "wouldn't it be nice if only it worked?" Good news—Andrew Milesk created an input driver called simply "Doubleclicker" (version 2.0, dated 2/4/90). It is a very effective alternative to the 1351/1351(a) COM mouse driver. When selected, it does what the name says: it double-clicks with a single press of the right button. This saves on wear and tear for the left button, and not to sound like a George Reeves "Superman" episode, but... you are able to leap into applications with a single click... it's a file, it's a driver, it's DOUBLE-CLICK! I myself have grown

so accustomed to it that I look for that function when using the computer at my workplace, and that ultramodern IBM doesn't do it! You'll find it a simple but handy feature to have.

All right, now you enter geoWrite, and realize that the font you need is tenth on the disk... nuts! Yet another shortcoming of geoWrite is the limited eight font accessibility. Now you have to exit your file and do the usual font-switch tango... or do you? Not if you get Payton W. Snider's Fontswap 1.0, which I mentioned briefly last issue. Not to insult anyone's intelligence, but again, the name says it all. This desk accessory is perfect for disks with large numbers of fonts and a user whose font choices vary with each new file. The first nine fonts on the disk are listed. If there are many more, they can be scrolled through one at a time, or page by page. You may toggle any seven, and when ready, select the melodramatic "DO IT!" icon. And it'll do it.

Now, let's hypothesize. You are a third-year college student and have over several dozen geoWrite files in your archives. The particular file you want is lost somewhere on a disk with a very large directory. But you need to find it quick! Let Nate Fiedler's 2k shareware jewel "FindFile" (version 1.0, 8/1/91) look for it! First of all, this is a desk accessory, so it can be used within any application including geoWrite. Then it has a very adept search function—enter the filename, press RETURN, and it will list its deskTop notepad's page number and file location number (1-6). Did you forget the file's name? No problem—the search function also uses wildcard characters such as the asterisk and question mark to help with your search. For example, if you enter "history *," FindFile will list up to the first nine files beginning with "history," be it "history American" to "history Zanzibarian" or even "history: Zzzzzzz." And with its "drive" option, drives A, B, C (and D if you can configure it) can be accessed. Each time you press the drive button, FindFile opens the next consecutive drive available. Once you have this desk accessory, you'll be able to sing, "it's a small disk after all..."

Large directories can be very intimidating, and even with such helpful tools as FindFile, it IS nice to have files neatly arranged on disk. Dir(ectory)Master version 1.0, by Kent L. Smotherman, is a powerful 4K application that can quickly arrange files in the order that you desire. What is most remarkable about this gem is its flexibility. The files can be categorically arranged by alphabet, reverse alphabet, size, type or date. What's more, you can toggle (select) all files on a disk, or just a few. The directory appears vertically, and so all files can be easily seen and selected. This means that even boot disks can be neatened up—just remember to deselect the boot files (they are rendered inoperative if they are moved from their first positions on a disk.)

Through the years I have increased my font library—some new ones I'd purchased from GEOS FontPaks or others downloaded from Q-Link joined older ones. But the occasion came when I needed a very unique font—one for a "history of English language" class. The font was named "I.P.A." for the International Phonetic Alphabet. It was done on Jim Collette's Font Editor 2.2 (10/8/88), and if you ever need to modify or create fonts, this is an excellent tool to do it with. Months before my new font was needed I'd received this application, not knowing its capabilities. Once the program is entered, you may either create a new font or adapt one on the current drive. It enables the user to edit the desired font by displaying one letter after another on a large pixel map. Point/click to select a pixel, and repeat to deselect. All modifications made to the letter appear WYSIWYG on the left side of the screen. You can therefore see how each letter will appear with each pixel you add or remove, and modify it accordingly.

It should be mentioned also that the Font Utilities that come with Creative Micro Designs' "Perfect Print LQ for GEOS" carry out a similar function in that new fonts can be created. Additionally the LQ fonts that result from those utilities are then usable with the high resolution Print System, which we will look at next issue.

Text file archives build up quickly over time, and you will eventually need to browse through some old writing samples. Sure enough, you will find an old file that will pique your curiosity, and upon opening it, it will upgrade from 1.3 to 2.1. Then you notice it is in the BSW font... and you'll wonder, WHICH font was it originally in? Here is where Dennis N. Seitz's "Identifont 3.2" (2/7/89) can assist you. Once selected, this application examines the text file or even the disk of your choice and determines the font(s) present. In the case of an unrecognizable font, a "hex." and "dec." number will be given.

Once your new file is done, your text is entered and you want to be sure that you met the essay requirements. Raymond A. Kerby's Copy Editor 1.0 (5/5/89) will do the job admirably. This application can run through a multi-page geoWrite document in seconds and gives the following data: total number of words in the document, total number of sentences and long words, average words per sentences, and a reader's index (approximate reading level). It is a fast and easy way of getting a quality check on finished writing.

Now, you have completed your text, with all the needed fonts identified, swapped, modified and or created. It is the proper length, and Copy Editor gives you a good review. Now I would like to direct you to Rick Krantz's Toolkit 1.2 (12/20/88) which is a file formatting wonder-worker. You know those LOOOOOONG documents that need every page to get a new font, different margins... takes some time without good help, doesn't it? Toolkit can perform entire document formatting at the touch of a button. Select "edit" mode, and the chosen file will be put into 40-column mode. "Print" puts the file into 80 column mode, and nicely adds paragraph indentations and one-inch margins at each "return." "Combine" can join two Write files together for more convenient printing, and "all fonts" will put the document into the font that is stored in the existing text scrap. These and other features make the Toolkit a must-have, and makes the finishing touch very quick indeed!

Voila! Your geoWrite file is now printer ready, and in less time than usual. Now for the next step—output improvement. In the next edition of geoSphere, we will look at some fonts, printer drivers and applications that will make even a nine pin dot matrix put out first rate material!

AMIGA

Repairs • Upgrades • Sales

Buy • Sell • Trade *Amiga* Hardware

64 USERS **SPECIALS** 128 USERS

5.25 SS/DD RECYCLED FLOPPY DISKS $49 CASE
5.25 DS/DD RECYCLED FLOPPY DISKS $59 CASE
CASE IS 1000 DISKS • ADD $11 S&H PER CASE
1200 BPS MODEMS 10/$29 + $6 S&H

Integrated Teknologies Inc

1101 Chestnut St • Suite A • Roselle • NJ 07203

FON 908 245-1313 FAX 908 245-9409

Hours Mon & Fri 1 - 6 PM Wed 1 - 8 PM

Moving?

Don't forget to notify COMMODORE WORLD!

Please call or write with your change of address 6 to 8 weeks prior to your move so that you won't miss a single issue!

CW Address Change, P.O. Box 646,
E. Longmeadow, MA 01028
413-525-0023

TECH STAR COMPUTER CENTER

(206) 251-9040

We'll repair your Commodore.

Serving the Commodore community since 1984.

REFURBISHED SPECIALS

SYSTEMS

C64........$69.00	1541......$69.00	C64 & 1541.......$120.00
64C........$79.00	1541 II...$79.00	64C & 1541 II....$140.00
C128......$89.00	1571......$89.00	C128 & 1571.....$150.00
VIC20.....$59.00	SX64.....$CALL	128D..................$CALL

PRINTERS

1525............$49.00	MPS 801......$59.00
1526............$69.00	MPS802.......$69.00
SG10...........$89.00	Star 10X.......$79.00
Star 15X......$99.00	SG10C.........$89.00
MPS 803.....$69.00	DPS 1101.....$99.00
DPS 1101....$99.00	OKI 10..........$69.00
OKI 20.........$79.00	

MONITORS

1701.........$119.00
1702.........$119.00
1802.........$119.00
1080.........$159.00
1084.........$159.00
1084S.......$159.00
CMONO...$ 29.00
MONO......$ 59.00

MISC.

Power supplies, cables, ribbons, paper, disks, cartridges, programs new and used, books, magazines, accessories.

DEMOS Part II

by Sherry Freedline

In the time since our first installment, I've found a few items to share with you. The most exciting item is the 4K Demo Contest sponsored by Driven magazine. This recent competition closed on July 1, 1996. Each entry was limited to one file of no more than $1000 bytes (4K), must be executable with the BASIC RUN command and, and had to be NTSC C-64 compatible. The prize was simply that of knowing your demo won the contest. With a contest of this nature the creators have the fun of participating in the contest, while the demo fans gain valuable additions to their demo collections. Competitions such as this help keep the scene alive.

Secondly, I stumbled across another magazine during one of my surfing expeditions. The magazine was produced by FOE (Forces of Evil) and is known as Coder's World. Three issues have been released, and the magazine is done in the style of a demo—similar to Driven magazine but with a different focus. Coder's World was introduced to the demo scene to provide the Commodore Community with a valuable reference guide for creating demos. I have enjoyed viewing demos for many years and the thought of learning how to create my own demo has crossed my mind many times—Coder's World provides folks such as myself with the tools to get started.

The most recent release of Coder's World, issue 3, contains articles demonstrating how to code 1x2 scrolls, character animation, playing music behind BASIC, logo swings, sprite scrolls, text fades, and more. Additionally, they've included an article on how to add class to your demos as well as revealing a few of their own machine language tips and tricks. Issue 3 also contains example programs demonstrating each of the routines explored within the issue.

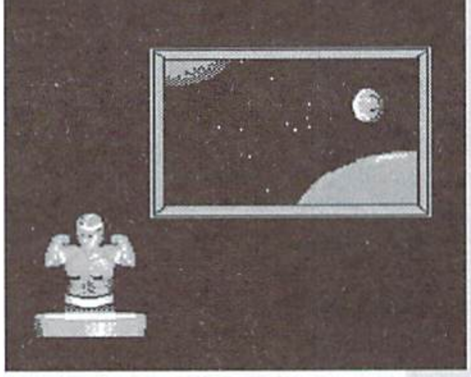
I think Coder's World was long overdue, and a very welcome addition to the demo scene. I applaud FOE (made up of The Phantom and Wrong Way) for their efforts and support. However, a word of warning is necessary due to the presence of some profanity in each issue of Coder's World. But, aside from that, it's a worthwhile read. Currently, FOE is working on an all new magazine called Solutions. Thanks FOE!

One more item I'd like to relay before moving along to the reviews is that of the explosion of demo sites on the Internet. I remember last year looking high and low in order to find "active" FTP Commodore demo sites. If you haven't been on the net lately, you'll be surprised at the number of places now available to satisfy all those demo urges. Almost every Commodore Demo group now has their own web page. From these web pages you can learn the history of the group, download their demos, and even view screen shots straight from their demos. The best feature of all is the creativity of their pages. Of course this can't yet be seen with a Commodore 64. We've always known these groups were creative, but now we get to see a whole new side of their creativity! Located elsewhere in this article you'll find a sidebar listing demo groups possessing their own web pages. So, while you're waiting for that next demo to be released, take some time to check out their sites. You'll be amazed at what you'll find!

Now it's finally time to move ahead to a few reviews. This issue's classic demos are Trap and Max Headroom. This issue's new soon-to-be classic demos are Digital Magic and Dawnfall.

Trap *Tony Crowthers and Ben Daglish*

The 1986 demo, Trap, is a demo of many names. You may also know it as the Space Movie or the Drummer Boy. The demo opens with a view of what looks like a native preparing to play a drum. In the upper right hand corner is a movie screen where the majority of the demo's action takes place.



The music starts, the native in the lower right hand corner begins to play his drum and the movie begins. As the demo moves along, you're treated to a visit from an astronaut (of course he's beamed down from his spaceship), a holographic style flashing face, and visits from other spaceships.

The whole demo lasts for more than eight minutes! All this comes from an amazingly small file of 43 blocks! At the end of the demo's movie are scrolling credits just like what you'd see at the real theater.

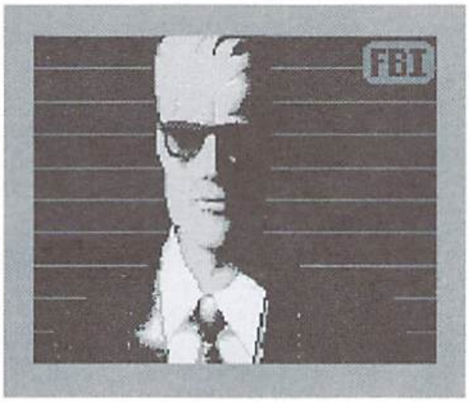
The demo's music reminds one of the movie 2001, creating an appropriate atmosphere for the space movie. Trap is a classic because it demonstrates all that can be done with the Commodore 64 and it does it well. And remember, this was still very early in the Commodore Demo

NTSC Demo Group Web Pages

Driven	http://soho.ios.com/~coolhnd
Electron	http://soho.ios.com/~coolhnd/tron.htm
FOE	http://www.cl.ais.net/wrongway/
Omni	http://hack.lakeheadu.ca/~revenge/
RPG	http://www.earthcom.net/~pegasus/rpg/
Style	http://test.dev-com.com/~interconnection/Style
Waveform/Millenium	http://marie.az.com/~waveform

PAL Demo Group Web Pages

Bonzai	http://chrom.imbg.ku.dk/sshbin/bonzai.html
Camelot	http://www.diku.dk/students/slammer/
Chalice	http://www.nada.kth.se/~d95-she/chalice
Delta System	http://www.inf.bme.hu/~mrc/
Equinoxe	http://rphc1.physik.uni-regensburg.de/~pem03049/eqx
Extend	http://www.lut.fi/~pora/extend/
Fairlight	http://www.ludd.luth.se/~watchman/fairlight/
Light	http://www.hh.se/stud/d94fa/light/light.html
Megastyle	http://login.nord.eunet.no/~scroll/msp.htm
Noice	http://www-und.ida.liu.se/~q95henra/
Padua	http://www.obh.snafu.de/~poing/padua/
Triad	http://www.df.lth.se/~triad/Triad_Home.html



era. Trap was definitely a demo well ahead of its time. Even back then folks knew the Commodore 64 would be capable of more than we could have ever dreamed.

Max Headroom *Brian Stephenson*
Next on my list of classics is the 1986 Max Headroom demo.

This demo is only partially famous because of Max Headroom. The main reason I've added it to my collection is because of the fantastic art work contained within this demo. The demo features no music, just a screen sized picture of Max opening and closing his mouth. But, the graphics are so well done that you may forget you are looking at a Commodore 64 screen.

After this demo was released, there were a few more Max Headroom demos appearing on the scene. Most memorable of them is the Max Headroom Swings demo in which smaller renditions of Max are accompanied by music.

Hmmm... I wonder if Max knows he's a Commodore Classic?

Dawnfall *by OXYRON*
Dawnfall is an extraordinary demo production resulting from the Party 95 demo competition. Throughout the entire demo one can't help but

notice the professional quality in which it is presented. The demo is contained solely within one file and moves along without any required interaction from the viewer.

Dawnfall's introductory screen reveals the letters of "Dawnfall" tumbling down upon an eclipse bearing a triangular Oxyron symbol. The letters then explode off into oblivion and the screen fades into the next portion of the introduction. Next the video type presentation zooms in and out on a scrolling Oxyron landscape. While you're busy marveling over the display, eerie music will hold you in suspense leading you to wonder what's in store for you next.

Naturally, now it's time for the opening demo credits. I loved the manner in which they are presented. The screen flashes and the names appear in large letters on your display until all the persons involved have been introduced. It's pretty hard to explain, but believe me, when you see it, you'll realize that it definitely adds to the overall professional quality of Dawnfall.

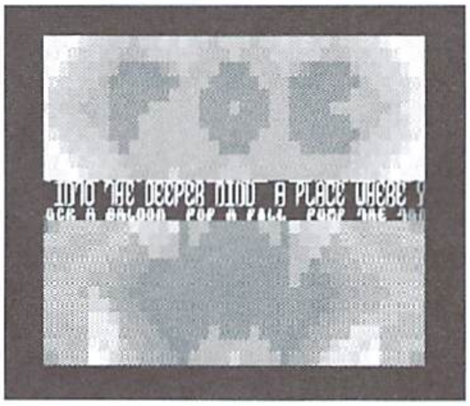
Now the fun begins as you are treated to a variety of three dimensional texture mapping routines. Included are a rotating chess board, a green, yellow and blue lattice design, a 3-D rotating upscroller and a few other items. The 3—D effects and the texture mapping are stunning! You'll easily forget that the graphics are displayed on a flat screen.

After the routines have decided they've truly astonished you, the music comes to a dead stop and the word "End" is flashed onto the screen. The music starts up again and, as in any normal video production, the end credits, thank you's and greetings roll up your screen. Behind the credits a plotter is hard at work holding your attention by designing a variety of shapes.

The demo ends with a teaser enticing you to find hidden pictures within the demo. To date I have found two of them. But, I've been told there are a total of three hidden pictures. The pictures are accessed by pressing a key or a combination of keys at the beginning, middle, and end of the demo. Unfortunately, Graham of Oxyron has asked me not to reveal the secrets. Are the secrets worth pursuing? Definitely!

Kudos to Oxyron, Jeff and Biz Kid of Camelot, and Sire of Lego for providing the Commodore Demoscene with such a high quality presentation—I hope we see more from them soon!

Digital Magic *by FOE*
FOE has really started 1996 off with a bang. Besides the above-mentioned Coder's World magazines, they have released three very impressive demos: S.E.T. (Smoke Exploration Tool), CAT (Copper Analysis Thingy), and, of course, Digital Magic. Each of these demos are so impressive that I had a very tough time selecting which one to review. And, just because I've selected Digital Magic, it doesn't mean the other



two aren't worthy of your time. If you fail to check them out, you'll be missing some very valuable additions to your collection!

Digital Magic is what the demo world calls a "mega-demo". It contains nine whole screens of graphics and music for you to enjoy! I'm going to highlight two of my favorites.

My first favorite display is on Digital Magic's fourth screen. FOE treats us to a page featuring eight sinus' waving through the screen. These sines place a total of 8,192 items on the screen! The colors are black and blue and leave the viewer in a nice relaxed state of mind. Of course, the whole mega-demo features outstanding music in addition to the cool graphic routines.

My other favorite screen is on page six of the demo. This page is known as Subliminal Colors. Here the creators, The Phantom and Wrong Way, have embedded the FOE logo into two animated color displays—one each the bottom and top halves of the screen. At first they may be hard to see. But keep looking because they are definitely there. It's a really cool effect that I haven't seen used in the past.

To reach each page of Digital Magic, you must press the space bar. What's unique about Digital Magic is that, first of all, the title page appears again each time you load a new page. Secondly, Digital Magic appears to be never ending due to the fact that once you've reached the final page it starts the demo over again.

As far as the other screens go, I'll leave them for you to explore. Because, after all, part of the fun of demos are the surprises contained within!

In Conclusion...
Before wrapping up this issue, I'd like to thank the members of FOE and Oxyron for taking the time to answer all my many questions while researching their respective demos. Also, I'd like to announce that all the demos reviewed here in Commodore World Magazine can be found on my web page: *http://www.lm.com/~qt/*.

That's it for this issue! See you next time!

Neural Network on a Commodore 64

by John Walker

When you first start to learn about computers, it's only natural to compare how a computer "thinks" with how people do. As you learn more, you tend to stop making such comparisons because you come to realize that far from the literal, dumb servants they're often pictured as in movies, computers are completely different from people both in how they operate and the kinds of problems they can solve.

But it's still fascinating to compare a computer to a human brain. Researchers are beginning to discover principles which may explain how the brain works, and while much of this research is extremely complicated and requires large, expensive computers, your Commodore 64 can be programmed in BASIC to mimic one fundamental part of brain function, pattern recognition. In doing this, your computer is not only doing something the brain does (and your reaction may be "I didn't know a computer could do that!"), it's doing it the brain's way—by simulating the neurons (brain cells) with which you think.

Computer Storage and Human Memory

First, let's compare what the word "memory" means in a computer as opposed to a human being. A computer memory is really a huge collection of pigeonholes into which numbers are stuffed. If you want to store something that isn't a number, such as a piece of text, a picture, or a sequence of musical notes, you first have to convert it to numbers, then put those numbers into the pigeonholes of the computer's memory. Machine language programming is just translating the instructions for solving a problem into numbers the computer can remember. A computer stores numbers.

Human memory is richer and much more complicated. What comes to mind when you look at the picture in Figure 1? Even though this picture is nothing more than 63 straight black lines on a white piece of paper, your brain immediately recognizes it as a sleeping cat. You can even imagine how a real cat would look if seen from other angles. The human brain seems to store and recall patterns. These patterns don't have to be pictures. You can recall pieces of music from only a few notes of the melody, think of words that rhyme with "frog" or that end with "ple", and quickly name the American presidents that have the same names as automobiles.

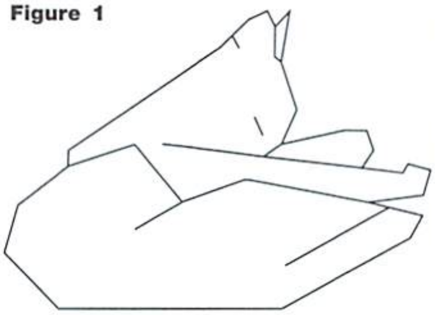
It isn't that human memory is better or worse than computer memory, but rather that they are entirely different things. Some computer scientists prefer the term "storage" to "memory" because it more accurately describes what the computer does. If you walk up to a computer and ask it "what is the name of the famous bridge in the same state as Disneyland" you won't have much success, but most people will immediately answer, "The Golden Gate Bridge". On the other hand, if you try to memorize a list of 20,000 two digit numbers, you probably won't succeed, yet a Commodore 64 can do this in less than a second and not make a single mistake. That computer and human memory are so different isn't all that surprising when you consider how differently constructed are the brain and a computer.

What Brains Are Made Of

The portion of the brain believed responsible for thought and memory consists primarily of nerve cells, or neurons. Each neuron has three parts, dendrites, a cell body, and an axon. The dendrites connect to the axons of other neurons. When these other neurons are stimulated, the dendrites convey the signal to the cell body via a synapse or connection, which either excites or inhibits the neuron (with a different strength for each synapse). When the excitation sufficiently outweighs inhibition, the neuron "fires". This sends a signal down its axon which in turn excites or inhibits other neurons, and perhaps causes a muscle to move.

Because neurons primarily connect to other neurons they form networks of great complexity. Figure 3 shows two fields of five neurons each, in which each neuron connects to every neuron in the other field. In this simple case we have 10 neurons with 5 connections (or synapses) each, for a total of 50 synapses. Now consider the brain. Researchers believe that the brain contains between ten and a hundred billion neurons, each of which connects to anywhere from a thousand to a hundred thousand other neurons, forming at least ten trillion connections, and probably far more. Compare this to the read-write memory of the Commodore 64, which is made up of about a quarter million transistors, and remember that each transistor is only a switch—far simpler than a neuron.

Figure 1



The Brain Simulator

Ten billion neurons, ten trillion connections: does it make sense to talk about simulating the brain on a computer? Can we make a computer recognize patterns the way a brain does? Remarkably, we can. A simple program can simulate the behavior of a network of interconnected neurons. You can show this program patterns and it will remember them. Then if you show it a similar pattern, it will find the pattern it has learned that is most like the pattern it is being shown. The technical name for this is an "associative memory", so called because it recalls items based on similarity, like the brain, as opposed to location, like a computer.

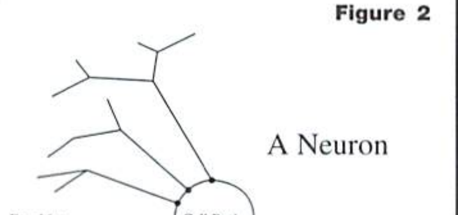
The Brain Simulator is written in BASIC for the 64 and 128 (in 64 mode). When you've finished typing in the program, save a copy to tape or disk. To run the program, simply load it and type RUN. You'll see two blank windows on the screen. Below them is a legend that explains the action of the function keys. When you type a letter or number, the dot pattern for that symbol appears in the left window. Try typing a few letters and numbers to see how this works. When you start the program, it doesn't know any patterns—so we'll teach it some. To learn a pattern, place it in the left window by pressing the key for the letter or number and then press F1. The program trains its simulated neurons to memorize the pattern (this takes about 30 seconds). READY reappears on the screen when the pattern has been learned. Go ahead and teach the program three different-looking letters, "A", "T", and "Z".

Now let's try recalling a pattern. Press the "A" key to place an A in the left window. Now press F3—this introduces errors in the pattern by randomly changing about 10% of the dots in the pattern each time you press it. After you press F3, you'll have a pattern that looks something like an A, but doesn't exactly match what we taught the program. Press F5 to start the recall process. The pattern is run back and forth through the neuron network until it stabilizes on a fixed pattern (an arrow in the middle of the screen shows the direction of the transfer). After the neuron network has "thought" about the problem for a few cycles, you'll probably get back the original A we taught the program (just like the brain, this process doesn't always remember the right thing; if the random changes made the pattern more like another pattern the program has learned, that one will be found instead).

Try entering T and Z, creating errors in them by pressing F3 one or two times, and recalling with F5. Note how the neuron network almost always recalls the correct pattern even though you've given it something quite different from what it learned. Enter "I" and try recalling with F5. The network recognizes this as T because I looks more like T than A or Z, the other patterns it has learned. This is what your brain does when it sees a pattern of lines and immediately thinks of a sleeping cat. Many researchers think the basic process the brain uses is much the same as the one used by this program.

You can make the program forget everything it has learned by pressing F4. If you press F6, the program exits to BASIC. When you leave the program everything it has learned is forgotten, but you can save learned patterns on the disk by pressing F7 and entering a file name. The next time you run the program you can reload the program's memory by pressing F8 and entering the same file name.

Figure 2

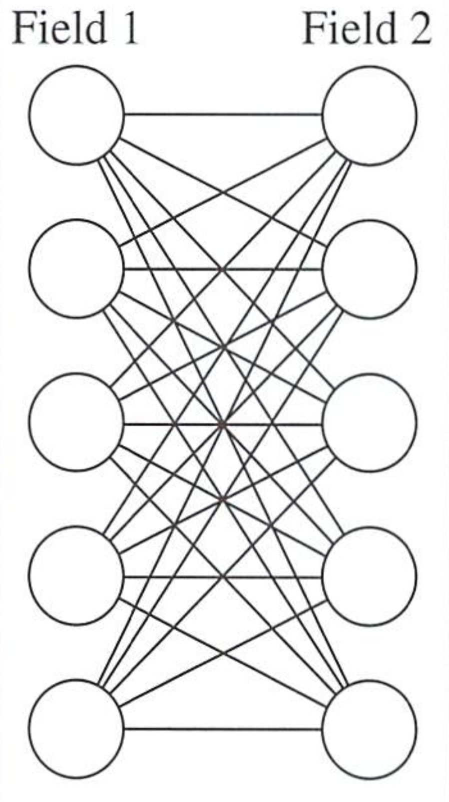


How it Works

The remarkable thing about this program is that it doesn't "know" it's recognizing letters and numbers. As far as the program is concerned, it could be learning phrases of music, combinations of medical symptoms, or pictures of animals.

The program simulates two fields of neurons with the arrays F1% and F2%, and displays these fields in the two windows on the screen. When you type a letter or number, the dot pattern for that symbol is read from the character ROM and stored in F1%. Lighted dots are stored as 1 and background dots as -1. The character patterns

Figure 3



are 6 dots wide by 7 dots high, so F1% and F2% both hold 42 numbers.

Each neuron in a field potentially connects to every neuron in the other field. Each connection, which is equivalent to a synapse in the brain, has,its own weight: positive to excite, negative to inhibit, and zero if there is no connection. These weights are stored in the 42x42 matrix M%, for a total of 1,764 connections. To learn a pattern, we form a matrix from the pattern in F1% and add it to the weight matrix M% (see lines 1020-1060 in the program). To recall a pattern we take the pattern in F1% and multiply it by the weight matrix (lines 1410-1480). If the value is 1 or more, we place a 1 in that position of F2%; if it's negative, we store a -1 there. If the value is zero, we leave the value in F2% alone. Then we take the value in F2% and multiply it back through the matrix, but swapping rows and columns, and store it back into F1% (lines 1540-1610). We keep this up until the pattern in F1% stops changing. That's our final value, the pattern we've recalled from the network.

Limits of Learning

The number of different patterns a neuron network can learn is determined by the number of neurons and connections. This very small network can learn only 2 or 3 distinct patterns before it begins to get confused. If you try to teach it 4 or 5 patterns, it will often recall the wrong pattern, or even a spurious pattern you never taught it. That's where the enormous complexity of the brain comes in; your brain has at least five billion times as many connections as this little program, so your ability to learn and remember is correspondingly greater.

But like a singing dog, it's not how well it sings but the fact it sings at all. With less than 250 lines of BASIC, we can make the Commodore 64 play the brain's game, pattern recognition, the brain's way, by simulating a neuron network. It can learn simply by being shown patterns and recall a similar pattern when shown something it's never seen before. Research into neuron networks is one of the most exciting and rapidly expanding fields in science today, bringing together computer science, psychology, mathematics, and biology to discover how the brain accomplishes the remarkable things it does. This research may lead to computers that can recognize faces, understand speech, and answer complicated questions. But more importantly, we may find answers to questions as old as mankind: "What is thought?", "What is memory?", and "How do people reason?". That your home computer can help you understand and experiment with such matters is testimony to the power latent within it.

For Further Information

This article only scratches the surface of the topic of neuron networks and associative memory. If you're interested in learning more, the following article and book are a good way to start:

Kosko, B., "Constructing an Associative Memory". Byte, September 1987.

Rumelhart, A., and McClelland, J., eds., Parallel Distributed Processing (two volumes), MIT Press, 1986.

```
√Σ   NEURAL.BAS

177  13 rem screen configuration
1Ø1  2Ø poke 5328Ø,Ø
117  3Ø poke 53281,Ø
218  4Ø print "{CLEAR/HOME}{WHT}";
166  5Ø open 15,8,15
214  6Ø rem variable declarations
16   7Ø dim f1%(42),f2%(42),m%(42,42)
29   8Ø dim v%,j,i
13   9Ø rem initialise screen
173  1ØØ print "{CLEAR/HOME}";
157  11Ø print "{4 SPACES}neuron network asso
     ciative memory"
16   12Ø print
5Ø   13Ø print "{HOME}{14 CRSR DN}";
76   14Ø print "f1 - teach pattern{5 SPACES}"
     ;
141  15Ø print "f2 - dump matrix"
122  16Ø print "f3 - randomize pattern ";
4    17Ø print "f4 - forget all"
185  18Ø print "f5 - recall pattern{4 SPACES}
     ";
117  19Ø print "f6 - quit"
217  2ØØ print "f7 - disc save{9 SPACES}";
154  21Ø print "f8 - disc load"
116  22Ø print
49   23Ø print "a-z, Ø-9: load pattern"
242  24Ø r1 = 4 : c1 = 5 : gosub 6ØØ
1Ø4  25Ø r1 = 4 : c1 = 25 : gosub 6ØØ
93   26Ø gosub 75Ø
11Ø  27Ø gosub 86Ø
126  28Ø gosub 97Ø:print " ready{4 SPACES}"
182  29Ø get a$ : if a$="" goto 29Ø
33   3ØØ gosub 97Ø:print "{1Ø SPACES}"
26   31Ø k=asc(a$)
1    32Ø ifa$>="Ø"anda$<="9"thenk=k+64:goto34
     Ø
29   33Ø if a$ < "a" or a$ > "z" then 5ØØ
2Ø5  34Ø gosub 97Ø:print "fetch ";a$
2Ø6  35Ø l%=Ø
198  36Ø k=(k-64)*8+53248
234  37Ø poke56333,127:poke 1,peek(1)and251
61   38Ø fori=Øto6:poke494Ø8+i,peek(k+i):next
211  39Ø poke 1,peek(1) or 4:poke 56333,129
248  4ØØ for i = Ø to 6
167  41Ø j% = peek(494Ø8+i)/2
7    42Ø for k=1 to 6
```

```
√Σ   NEURAL.BAS (cont.)

157  43Ø l%=l%+1
61   44Ø f1%(l%) = -1 + (2 * (j% and 1))
178  45Ø j%=j%/2
23Ø  46Ø next k
232  47Ø next i
148  48Ø gosub 75Ø : gosub 86Ø : goto 28Ø
154  49Ø rem dispatch function key commands
166  5ØØ j%=asc(a$)-132
2Ø7  51Ø if j%=1 then gosub 1ØØØ:goto 28Ø
25   52Ø if j%=5 then gosub 1Ø8Ø:goto 9Ø
53   53Ø if j%=2 then gosub 121Ø:goto 28Ø
141  54Ø if j%=6 then gosub 168Ø:goto 28Ø
172  55Ø if j%=3 then gosub 129Ø:goto 28Ø
38   56Ø if j%=7 then print "";:close15:end
66   57Ø if j%=4 then gosub 18ØØ:goto 9Ø
195  58Ø if j%=8 then gosub 199Ø:goto 9Ø
193  59Ø go to 28Ø
35   6ØØ rem draw borders for fields
175  61Ø for i=Ø to 1
12   62Ø v=1Ø24+4Ø*(r1+(i*8))+c1
2Ø7  63Ø poke v,112+(-3*i)
249  64Ø for j=1 to 8
158  65Ø poke v+j,67
169  66Ø next j
248  67Ø poke v+9,11Ø+(15*i)
191  68Ø next i
22   69Ø for i=1 to 7
62   7ØØ v=1Ø24+4Ø*(r1+i)+c1
118  71Ø poke v,93
134  72Ø poke v+9,93
237  73Ø next i
113  74Ø return
222  75Ø rem update field f2% on screen
98   76Ø l%=Ø
1Ø4  77Ø for i=Ø to 6
38   78Ø v% = 1Ø24+4Ø*(i+5)+6
14Ø  79Ø for j=2 to 7
5    8ØØ l%=l%+1
148  81Ø iff1%(l%)=1thenpokev%+(8-j),81:goto8
     3Ø
77   82Ø poke v%+(8-j),32
8Ø   83Ø next j
88   84Ø next i
227  85Ø return
19Ø  86Ø rem update field f1% on screen
212  87Ø l%=Ø
```

```
√Σ   NEURAL.BAS (cont.)

218  88Ø for i = Ø to 6
2    89Ø v%=1Ø24+4Ø*(i+5)+26
254  9ØØ for j=2 to 7
123  91Ø l%=l%+1
233  92Ø if f2%(l%)=1 then poke v%+(8-j),81:g
     oto 94Ø
219  93Ø poke v%+(8-j),32
198  94Ø next j
206  95Ø next i
74   96Ø return
6    97Ø rem position to status area
229  98Ø print "{HOME}{22 CRSR DN}";
104  99Ø return
164  1ØØØ rem train on pattern in f1%
60   1Ø1Ø gosub 97Ø:print "training"
190  1Ø2Ø for i = 1 to 42
178  1Ø3Ø for j = 1 to 42
81   1Ø4Ø m%(i,j)=m%(i,j)+f1%(i)*f1%(j)
53   1Ø5Ø next j
61   1Ø6Ø next i
184  1Ø7Ø return
162  1Ø8Ø rem print part of matrix
143  1Ø9Ø print "{CLEAR/HOME}";
11   11ØØ for i=1 to 24
52   111Ø for j=1 to 39
241  112Ø ifm%(i,j)<Øthenprint "";:goto114Ø
159  113Ø print "";
200  114Ø print chr$(asc("Ø")+abs(m%(i,j)));
145  115Ø next j
36   116Ø print
163  117Ø next i
215  118Ø print "press any key to continue:";
35   119Ø get a$ : if a$="" goto 119Ø
59   12ØØ return
11   121Ø rem randomise 1Ø percent of f1%
57   122Ø gosub 97Ø:print "random"
139  123Ø for i=1 to 42
245  124Ø if rnd(Ø) > Ø.1 then 126Ø
220  125Ø f1%(i)=-f1%(i)
6    126Ø next i
87   127Ø gosub 75Ø
139  128Ø return
186  1293 rem recall from pattern
236  13ØØ gosub 97Ø:print "recall"
248  131Ø p%=1Ø24+4Ø*9+19
167  132Ø rem initially copy f1 to f2
4    133Ø poke p%+1,asc("=")
253  134Ø for i=1 to 42
206  135Ø f2%(i)=f1%(i)
98   136Ø next i
190  137Ø gosub 86Ø
159  138Ø rem f1 to f2 pass
38   139Ø poke p%,asc("=")
107  14ØØ poke p%+2,asc(">")
52   141Ø for j=1 to 42
11   142Ø v%=Ø
66   143Ø for i=1 to 42
61   144Ø v%=v%+f1%(i)*m%(i,j)
184  145Ø next i
65   146Ø v%=sgn(v%)
36   147Ø if v%<>Ø then f2%(j)=v%
216  148Ø next j
57   149Ø gosub 86Ø
227  15ØØ rem f2 to f1 pass
64   151Ø c%=Ø
188  152Ø poke p%,asc("<")
243  153Ø poke p%+2,asc("=")
180  154Ø for i=1 to 42
141  155Ø v%=Ø
222  156Ø for j=1 to 42
170  157Ø v%=v%+f2%(j)*m%(i,j)
```

```
√Σ   NEURAL.BAS (cont.)

69   158Ø next j
195  159Ø v%=sgn(v%)
7    16ØØ ifv%<>Øandv%<>f1%(i)thenf1%(i)=v%:c
     %=1
89   161Ø next i
166  162Ø gosub 75Ø
139  163Ø if c%<>Ø goto 138Ø
84   164Ø poke p%,asc(" ")
34   165Ø poke p%+1,asc(" ")
83   166Ø poke p%+2,asc(" ")
19   167Ø return
231  168Ø rem forget all - clear memory
27   169Ø gosub 97Ø:print "forget"
75   17ØØ for i=1 to 42
55   171Ø f1%(i)=Ø
35   172Ø f2%(i)=Ø
107  173Ø for j=1 to 42
229  174Ø m%(i,j)=Ø
235  175Ø next j
243  176Ø next i
77   177Ø gosub 75Ø
90   178Ø gosub 86Ø
139  179Ø return
120  18ØØ rem save state to disc file
243  181Ø gosub 97Ø:print "save"
106  182Ø print "{CLEAR/HOME}";
134  183Ø input "file name: ";a$
122  184Ø a$="@Ø:"+a$+",s,w"
81   185Ø open 5,8,5,a$
4    186Ø for i=1 to 42:print#5,f1%(i):next
159  187Ø gosub 224Ø
91   188Ø for i=1 to 42:print#5,f2%(i):next
163  189Ø gosub 224Ø
50   19ØØ for i=1 to 42
62   191Ø for j=1 to 42
101  192Ø print#5,m%(i,j)
168  193Ø next j
209  194Ø gosub 224Ø
186  195Ø next i
186  196Ø close 5
232  197Ø print "";
66   198Ø return
210  199Ø rem restore state from disc file
122  2ØØØ gosub 97Ø:print "restore"
19   2Ø1Ø print "{CLEAR/HOME}";
69   2Ø2Ø input "file name: ";a$
141  2Ø3Ø a$="@Ø:"+a$+",s,r"
227  2Ø4Ø p%=asc("m")
66   2Ø5Ø gosub 224Ø
2    2Ø6Ø open 5,8,5,a$
220  2Ø7Ø for i=1 to 42
95   2Ø8Ø input#5,f1%(i)
55   2Ø9Ø next i
112  21ØØ gosub 224Ø
228  211Ø for i=1 to 42
106  212Ø input#5,f2%(i)
111  213Ø next i
168  214Ø gosub 224Ø
15   215Ø for i=1 to 42
59   216Ø for j=1 to 42
72   217Ø input#5,m%(i,j)
147  218Ø next j
222  219Ø gosub 224Ø
165  22ØØ next i
165  221Ø close 5
51   222Ø return
80   223Ø rem disc error check
136  224Ø input#15,en,em$,et,es
218  225Ø if en>Øthen print en,em$,et,es:stop
107  226Ø return
```

A 6502 Programmer's Introduction to the 65816

by Brett Tabke

After programming in 6502 machine language for over a decade, I was getting a bit BORED. One can only code the same routines with the same opcodes so many times before the nausea of repetition becomes overpowering. When I heard the news that CMD was building a cartridge based on a 20 MHz 65816 I was overjoyed. For years I've heard those with 65816 based systems brag about its capabilities. To us old 6502 programmers, the opportunity to program the fabled 65816 is a new lease on life.

The 65816 is an 8-/16-bit register selectable upgrade to the 6502 series processor. With 24 bit addressing of up to 16 Megabytes of RAM, the powerful 65816 is a logical upgrade that leaves 6502 programmers feeling right at home. It is amazing how fast one can adapt to the new processor. It sounds funny to say it, but the only difficulty I have had learning the 65816 is that there are so many options and choices to complete the same task, that it is hard to decide which method is best.

To get started programming the 65816, I would recommend purchasing the book, "Programming the 65816" from The Western Design Center, manufacturer of the 65816. While it is a bit pricey, the sheer quality and content of the 600 page book is worth the money. Rarely, if ever, has there been a CPU manual as thorough and detailed as the Western Design book. If you know 6502 assembly, then Programming the 65816 is probably the only 65816 book you will ever need.

Getting a Feel for the Modes

The 65816 may be operated in Native mode or 6502 Emulation mode. Emulation mode is a 100% 6502 compatible mode where the whole processor looks and feels like a vintage 6502. Native mode offers 8- or 16-bit user registers and full access to 24-bit addressing.

While in emulation mode, not only are all the 6502 opcodes present in their virgin form, but the new 65816 instructions are also available for usage. In fact, the first lesson to learn about programming the 65816 is that emulation mode is much more powerful than a stock 6502. The only true difference between emulation mode and our venerable C64's 6510 processor is that unimplemented opcodes will not produce the results expected on the former. Since all 256 of the potential opcodes are now implemented on the 65816, older C64 software that uses previously unimplemented opcodes will produce erratic results.

To select between emulation and native modes, a new phantom hidden emulation bit (E) was added to the status register. Shown in programming models hanging on top of the Carry bit, the emulation bit is only accessible by one instruction. The new instruction (XCE) exchanges the status of the Carry bit and Emulation bit. To move to emulation mode, set the carry and issue an XCE instruction. To move to native mode, clear the carry and issue the XCE instruction.

My, How Your Index Registers Have Grown!

While in native mode there are two new directly accessible bits present in the status register. The 65816 implements new hardware interrupt vectors which include a new hardware BRK vector in ROM; therefore, the old BRK bit of the status register is no longer needed. The BRK bit is replaced with the X bit to select either 8- or 16-bit index registers. The former empty bit 5 is now filled with the M bit to specify the accumulator and memory access as 8- or 16-bit.

Two new instructions are used to clear or set bits within the status register. The SEP instruction sets bits, and REP clears bits. SEP and REP use a one byte immediate addressing mode operand to specify which bits are to be set or cleared. For example, to set the X bit for 8 bit user registers:

```
SEP #%00010000 ; set bit 4 for 8-bit index
               ; registers.
```

Or to clear bit 4:

```
REP #%00010000 ; clear bit 4 for 16-bit index
               ; registers.
```

When in 8 bit mode, the index registers perform their function in standard 6502 form. When status bit X is set to 0, both the X and Y index registers become 16 bits wide. With a 16-bit index register you can now reach out to a full 64K with the various indexed addressing modes. An absolute load to an index register in 16-bit mode will retrieve 2 bytes of memory—the one at the effective address and the one at the effective address plus one. Simple things like INX or DEY work on a full 16 bits, which means you no longer have to specify a memory location for various counters, and loops based on index counters can now be coded in a more efficient manner.

The formerly empty status register bit 5 is now referred to as bit M. M is used to specify an 8- or 16-bit wide accumulator and memory accesses. When in 8 bit mode, (M=1), the high order 8 bits are still accessible by exchanging the low and high bytes with a XBA instruction—it is like having two accumulators! However; when set for a full 16-bit wide accumulator, all math and accumulator oriented logical instructions operate on all 16 bits! If you add up the clock cycles and bytes required to perform a standard two byte addition, you can start to see the true power of 16-bit registers.

More Register Improvements

Zero Page has now been renamed to Direct Page—corporate thinking, go figure. A new processor register D was added to allow Direct Page to be moved anywhere within the first 64K of memory. The direct page register is 16 bits wide, so you can now specify the start of direct page at any byte. Several old instructions now include direct page addressing as well. To move direct page, just push the new value onto the stack (16 bits) and then PLD to pull it into the direct page register. You may also transfer the value from the 16-bit accumulator to the direct page register with the TCD instruction. Direct page may also be moved while in emulation mode.

While in native mode, the stack pointer is a full 16 bits wide, which means the stack is no longer limited to just 256 bytes. It can be moved anywhere within the first 64K of memory (although while in emulation mode, the stack is located at page one). There are several new addressing modes that can use the stack pointer as a quasi-index register to access memory. Numerous new push and pull instructions allow you to manipulate the stack. A few of the more useful stack instructions useful to programmers, are the new instructions to push & pull index registers with PHX/PHY and PLX/PLY.

Two other new processor registers are the Program Bank Register (PBR) and the Data Bank Register (DBR). The Program Bank Register can be thought of as extending the program counter out to 24 bits. Although you can JSR and JMP to routines located in other RAM banks, individual routines on the 65816 still must run within a single bank of 64K—there's no automatic rollover from one bank of RAM to the next when executing successive instructions. In this sense, it may help to think of the 65816 processor as a marriage of Commodore's C128 Memory Management Unit (MMU) and an 'enhanced' 6502—a very similar concept.

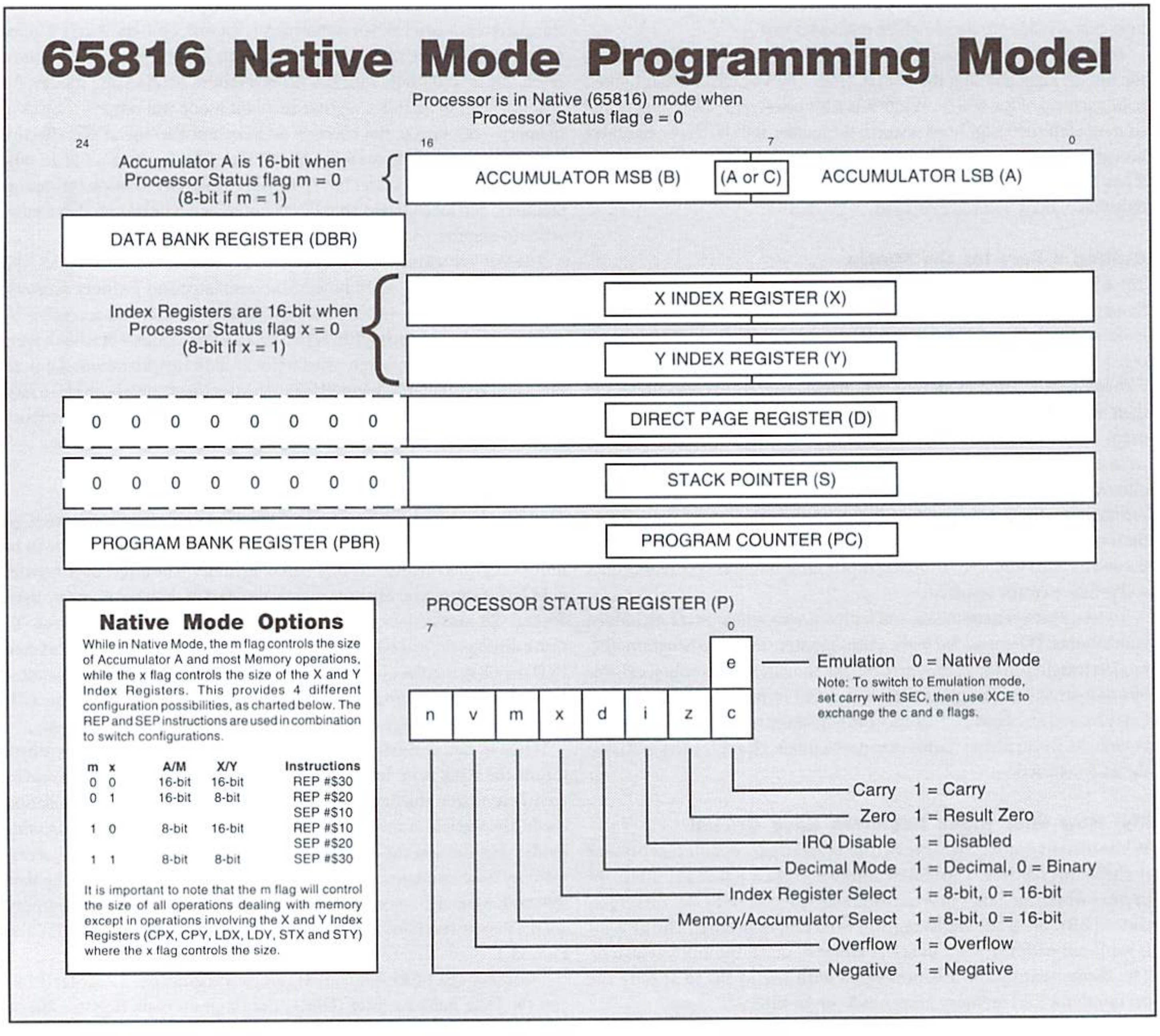
The Data Bank Register is used to reach out to any address within the 16 megabyte address space of the 65816. When any of the addressing modes that specify a 16-bit address are used, the Data Bank byte is appended to the instruction address. This allows access to all 16 megabytes without having to resort to 24-bit addressing instruction, and helps enable code that can operate from any bank.

New Addressing Modes

There are nine new addressing modes on the 65816. Several new instructions are designed to help create relocatable code that can execute at any address. The use of relocatable code on the 6502 was extremely limited. With 16 megabytes of address space, writing relocatable code increases the overall utility of the program. To write relocatable code, several new instructions use Program Counter Relative Long addressing. This allows relative branching within a 64K bank of RAM. There's also Stack Relative addressing, and a push instruction to place the program counter onto the stack, so that a code fragment can pull it back off and can instantly know its execution address.

Another new feature are two Block Move instructions, one for forward MVP and one for backward MVN. Simply load the 16-bit X register with the starting address, the Y index register with the ending address, the accumulator with the number of bytes to move, and issue the MVP or MVN instructions. MVN is for move negative, and MVP is for move positive, so that your moves don't overwrite themselves. Block Moves use two operand bytes: one for the source bank of 64K and one for the destination bank. Memory is moved at the rate of seven clock cycles per byte.

Several new addressing modes are used to access the full address space. A 65816 assembler would decode "long" addressing given this input:



65816 Native Mode Programming Model
Processor is in Native (65816) mode when Processor Status flag e = 0
24
16
7
0
Accumulator A is 16-bit when Processor Status flag m = 0 (8-bit if m = 1)
ACCUMULATOR MSB (B)
(A or C)
ACCUMULATOR LSB (A)
DATA BANK REGISTER (DBR)
Index Registers are 16-bit when Processor Status flag x = 0 (8-bit if x = 1)
X INDEX REGISTER (X)
Y INDEX REGISTER (Y)
0 0 0 0 0 0 0 0
DIRECT PAGE REGISTER (D)
0 0 0 0 0 0 0 0
STACK POINTER (S)
PROGRAM BANK REGISTER (PBR)
PROGRAM COUNTER (PC)
PROCESSOR STATUS REGISTER (P)
7
0
e
n v m x d i z c
Emulation 0 = Native Mode
Note: To switch to Emulation mode, set carry with SEC, then use XCE to exchange the c and e flags.
Carry 1 = Carry
Zero 1 = Result Zero
IRQ Disable 1 = Disabled
Decimal Mode 1 = Decimal, 0 = Binary
Index Register Select 1 = 8-bit, 0 = 16-bit
Memory/Accumulator Select 1 = 8-bit, 0 = 16-bit
Overflow 1 = Overflow
Negative 1 = Negative

Native Mode Options

While in Native Mode, the m flag controls the size of Accumulator A and most Memory operations, while the x flag controls the size of the X and Y Index Registers. This provides 4 different configuration possibilities, as charted below. The REP and SEP instructions are used in combination to switch configurations.

m	x	A/M	X/Y	Instructions
0	0	16-bit	16-bit	REP #$30
0	1	16-bit	8-bit	REP #$20 SEP #$10
1	0	8-bit	16-bit	REP #$10 SEP #$20
1	1	8-bit	8-bit	SEP #$30

It is important to note that the m flag will control the size of all operations dealing with memory except in operations involving the X and Y Index Registers (CPX, CPY, LDX, LDY, STX and STY) where the x flag controls the size.

```
LDA $0445F2    ; load byte from $45F2 of RAM
               ; bank 4

LDA $03412F,x  ; load byte from $412F of bank 3
               ; plus x.
```

Quite a few instructions have been given new addressing modes. How many times have you wanted to do this:

```
LDA ($12)      ; load indirect without an
               ; offset.
```

Or how about a table of routine addresses:

```
JSR ($1234,x)  ; jump to a subroutine via
               ; indexed indirect addressing!
```

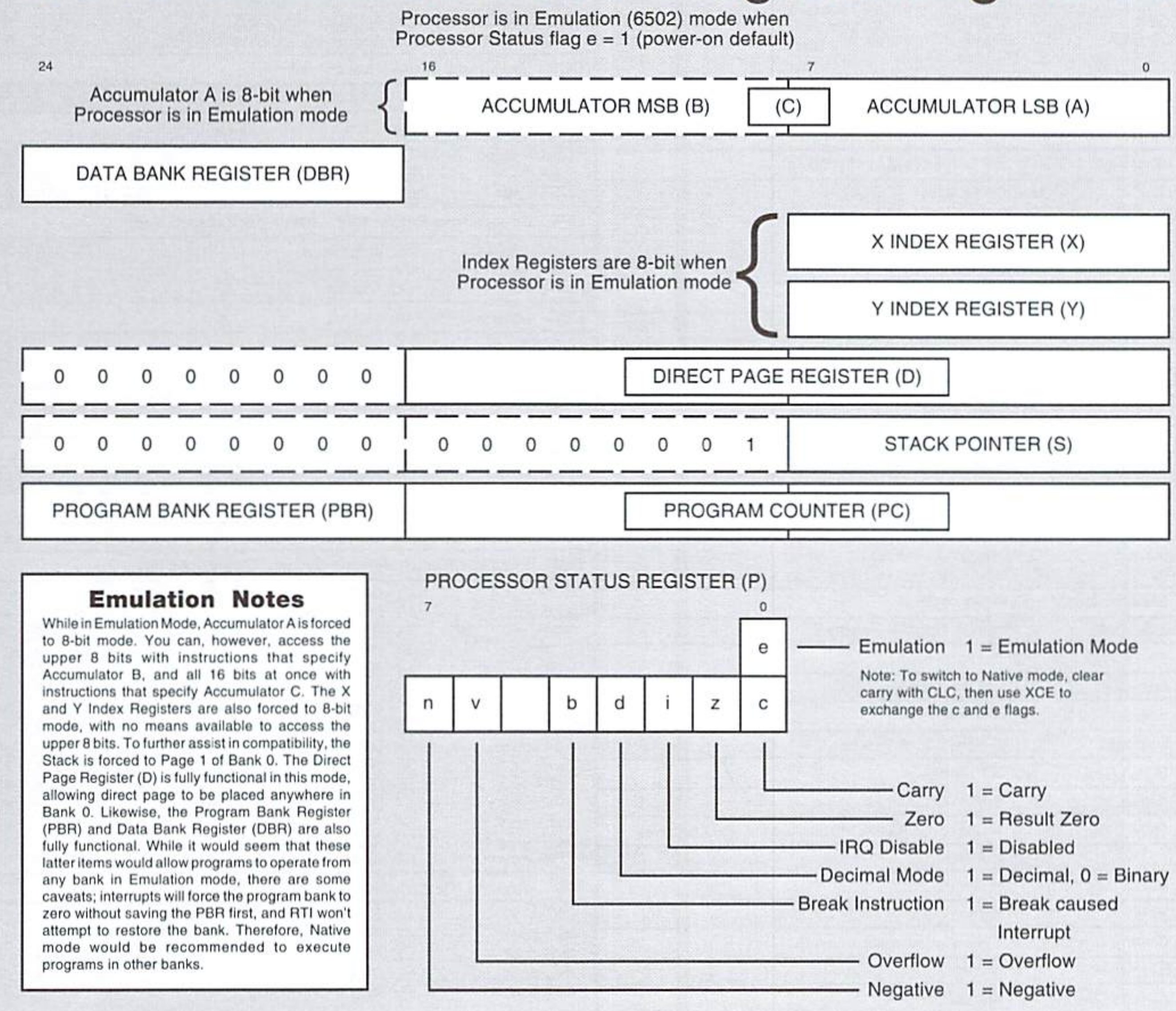
Other fun new instructions:

TXY,TYX	Transfer directly between index registers
BRA	Branch always regardless of status bits
TSB	Test and set any bit of a byte
TRB	Test and reset (clear) any bit of a byte
INC A/DEC A	Increment or decrement the accumulator directly
STZ	Store a zero to any byte

Summing Up

As you can see, the 65816 opens up a whole new world of programming—it feels like a new lease on life. Of course, it's going to take some time to learn the new processor. But while the 20 MHz speed is a nice perk, I believe that the real power of CMD's new peripheral is indeed the engine under its hood: the 65816—a *super* CPU!

65816 Emulation Mode Programming Model



Processor is in Emulation (6502) mode when
Processor Status flag e = 1 (power-on default)
24
16
7
0
Accumulator A is 8-bit when Processor is in Emulation mode
ACCUMULATOR MSB (B)
(C)
ACCUMULATOR LSB (A)
DATA BANK REGISTER (DBR)
Index Registers are 8-bit when Processor is in Emulation mode
X INDEX REGISTER (X)
Y INDEX REGISTER (Y)
0 0 0 0 0 0 0 0
DIRECT PAGE REGISTER (D)
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1
STACK POINTER (S)
PROGRAM BANK REGISTER (PBR)
PROGRAM COUNTER (PC)
PROCESSOR STATUS REGISTER (P)
7
0
e
n v b d i z c
Emulation 1 = Emulation Mode
Note: To switch to Native mode, clear carry with CLC, then use XCE to exchange the c and e flags.
Carry 1 = Carry
Zero 1 = Result Zero
IRQ Disable 1 = Disabled
Decimal Mode 1 = Decimal, 0 = Binary
Break Instruction 1 = Break caused Interrupt
Overflow 1 = Overflow
Negative 1 = Negative

Emulation Notes

While in Emulation Mode, Accumulator A is forced to 8-bit mode. You can, however, access the upper 8 bits with instructions that specify Accumulator B, and all 16 bits at once with instructions that specify Accumulator C. The X and Y Index Registers are also forced to 8-bit mode, with no means available to access the upper 8 bits. To further assist in compatibility, the Stack is forced to Page 1 of Bank 0. The Direct Page Register (D) is fully functional in this mode, allowing direct page to be placed anywhere in Bank 0. Likewise, the Program Bank Register (PBR) and Data Bank Register (DBR) are also fully functional. While it would seem that these latter items would allow programs to operate from any bank in Emulation mode, there are some caveats; interrupts will force the program bank to zero without saving the PBR first, and RTI won't attempt to restore the bank. Therefore, Native mode would be recommended to execute programs in other banks.

Guide to 6502/65C02/65816 Instructions

Assembler Example	HEX	Addressing Mode	02	C02	816	Bytes	Cycles
ADC *Add With Carry* [Flags affected: n,v,z,c]							
ADC (*dp*,X)	61	DP Indexed Indirect,X	√	√	√	2	$6^{1,2,4}$
ADC *sr*,S	63	Stack Relative			√	2	$4^{1,4}$
ADC *dp*	65	Direct Page	√	√	√	2	$3^{1,2,4}$
ADC [*dp*]	67	DP Indirect Long			√	2	$6^{1,2,4}$
ADC #*const*	69	Immediate	√	√	√	2^{17}	$2^{1,4}$
ADC *addr*	6D	Absolute	√	√	√	3	$4^{1,4}$
ADC *long*	6F	Absolute Long			√	4	$5^{1,4}$
ADC (*dp*),Y	71	DP Indirect Indexed,Y	√	√	√	2	$5^{1,2,3,4}$
ADC (*dp*)	72	DP Indirect		√	√	2	$5^{1,2,4}$
ADC (*sr*,S),Y	73	SR Indirect Indexed,Y			√	2	$7^{1,4}$
ADC *dp*,X	75	DP Indexed,X	√	√	√	2	$4^{1,2,4}$
ADC [*dp*],Y	77	DP Indirect Long Indexed,Y			√	2	$6^{1,2,4}$
ADC *addr*,Y	79	Absolute Indexed,Y	√	√	√	3	$4^{1,3,4}$
ADC *addr*,X	7D	Absolute Indexed,X	√	√	√	3	$4^{1,3,4}$
ADC *long*,X	7F	Absolute Long Indexed,X			√	4	$5^{1,4}$
AND *AND Accumulator With Memory* [Flags affected: n,z]							
AND (*dp*,X)	21	DP Indexed Indirect,X	√	√	√	2	$6^{1,2}$
AND *sr*,S	23	Stack Relative			√	2	4^{1}
AND *dp*	25	Direct Page	√	√	√	2	$3^{1,2}$
AND [*dp*]	27	DP Indirect Long			√	2	$6^{1,2}$
AND #*const*	29	Immediate	√	√	√	2^{17}	2^{1}
AND *addr*	2D	Absolute	√	√	√	3	4^{1}
AND *long*	2F	Absolute Long			√	4	5^{1}
AND (*dp*),Y	31	DP Indirect Indexed,Y	√	√	√	2	$5^{1,2,3}$
AND (*dp*)	32	DP Indirect		√	√	2	$5^{1,2}$
AND (*sr*,S),Y	33	SR Indirect Indexed,Y			√	2	7^{1}
AND *dp*,X	35	DP Indexed,X	√	√	√	2	$4^{1,2}$
AND [*dp*],Y	37	DP Indirect Long Indexed,Y			√	2	$6^{1,2}$
AND *addr*,Y	39	Absolute Indexed,Y	√	√	√	3	$4^{1,3}$
AND *addr*,X	3D	Absolute Indexed,X	√	√	√	3	$4^{1,3}$
AND *long*,X	3F	Absolute Long Indexed,X			√	4	5^{1}
ASL *Accumulator or Memory Shift Left* [Flags affected: n,z,c]							
ASL *dp*	06	Direct Page	√	√	√	2	$5^{2,5}$
ASL A	0A	Accumulator	√	√	√	1	2
ASL *addr*	0E	Absolute	√	√	√	3	6^{5}
ASL *dp*,X	16	DP Indexed,X	√	√	√	2	$6^{2,5}$
ASL *addr*,X	1E	Absolute Indexed,X	√	√	√	3	$7^{5,6}$
BCC *Branch if Carry Clear* [Flags affected: none] [Alias: BLT]							
BCC *nearlabel*	90	Program Counter Relative	√	√	√	2	$2^{7,8}$
BCS *Branch if Carry Set* [Flags affected: none] [Alias: BGE]							
BCS *nearlabel*	B0	Program Counter Relative	√	√	√	2	$2^{7,8}$
BEQ *Branch if Equal* [Flags affected: none]							
BEQ *nearlabel*	F0	Program Counter Relative	√	√	√	2	$2^{7,8}$
BIT *Test Bits* [Flags affected: z (immediate mode) n,v,z (non-immediate modes)]							
BIT *dp*	24	Direct Page	√	√	√	2	$3^{1,2}$
BIT *addr*	2C	Absolute	√	√	√	3	4^{1}
BIT *dp*,X	34	DP Indexed,X		√	√	2	$4^{1,2}$
BIT *addr*,X	3C	Absolute Indexed,X		√	√	3	$4^{1,3}$
BIT #*const*	89	Immediate		√	√	2^{17}	2^{1}
BMI *Branch if Minus* [Flags affected: none]							
BMI *nearlabel*	30	Program Counter Relative	√	√	√	2	$2^{7,8}$
BNE *Branch if Not Equal* [Flags affected: none]							
BNE *nearlabel*	D0	Program Counter Relative	√	√	√	2	$2^{7,8}$
BPL *Branch if Plus* [Flags affected: none]							
BPL *nearlabel*	10	Program Counter Relative	√	√	√	2	$2^{7,8}$
BRA *Branch Always* [Flags affected: none]							
BRA *nearlabel*	80	Program Counter Relative		√	√	2	3^{8}
BRK *Break* [Flags affected: b,i (6502) b,d,i (65C02/65816 Emulation) d,i (65816 Native)]							
BRK	00	Stack/Interrupt	√	√	√	2^{18}	7^{9}
BRL *Branch Long Always* [Flags affected: none]							
BRL *label*	82	Program Counter Relative Long			√	3	4
BVC *Branch if Overflow Clear* [Flags affected: none]							
BVC *nearlabel*	50	Program Counter Relative	√	√	√	2	$2^{7,8}$
BVS *Branch if Overflow Set* [Flags affected: none]							
BVS *nearlabel*	70	Program Counter Relative	√	√	√	2	$2^{7,8}$

Assembler Example	HEX	Addressing Mode	02	C02	816	Bytes	Cycles
CLC *Clear Carry* [Flags affected: c]							
CLC	18	Implied	√	√	√	1	2
CLD *Clear Decimal Mode Flag* [Flags affected: d]							
CLD	D8	Implied	√	√	√	1	2
CLI *Clear Interrupt Disable Flag* [Flags affected: i]							
CLI	58	Implied	√	√	√	1	2
CLV *Clear Overflow Flag* [Flags affected: v]							
CLV	B8	Implied	√	√	√	1	2
CMP *Compare Accumulator With Memory* [Flags affected: n,z,c]							
CMP (*dp*,X)	C1	DP Indexed Indirect,X	√	√	√	2	$6^{1,2}$
CMP *sr*,S	C3	Stack Relative			√	2	4^{1}
CMP *dp*	C5	Direct Page	√	√	√	2	$3^{1,2}$
CMP [*dp*]	C7	DP Indirect Long			√	2	$6^{1,2}$
CMP #*const*	C9	Immediate	√	√	√	2^{17}	2^{1}
CMP *addr*	CD	Absolute	√	√	√	3	4^{1}
CMP *long*	CF	Absolute Long			√	4	5^{1}
CMP (*dp*),Y	D1	DP Indirect Indexed,Y	√	√	√	2	$5^{1,2,3}$
CMP (*dp*)	D2	DP Indirect		√	√	2	$5^{1,2}$
CMP (*sr*,S),Y	D3	SR Indirect Indexed,Y			√	2	7^{1}
CMP *dp*,X	D5	DP Indexed,X	√	√	√	2	$4^{1,2}$
CMP [*dp*],Y	D7	DP Indirect Long Indexed,Y			√	2	$6^{1,2}$
CMP *addr*,Y	D9	Absolute Indexed,Y	√	√	√	3	$4^{1,3}$
CMP *addr*,X	DD	Absolute Indexed,X	√	√	√	3	$4^{1,3}$
CMP *long*,X	DF	Absolute Long Indexed,X			√	4	5^{1}
COP *Co-Processor Enable* [Flags affected: d,i]							
COP *const*	02	Stack/Interrupt			√	2^{18}	7^{9}
CPX *Compare Index Register X with Memory* [Flags affected: n,z,c]							
CPX #*const*	E0	Immediate	√	√	√	2^{19}	2^{10}
CPX *dp*	E4	Direct Page	√	√	√	2	$3^{2,10}$
CPX *addr*	EC	Absolute	√	√	√	3	4^{10}
CPY *Compare Index Register Y with Memory* [Flags affected: n,z,c]							
CPY #*const*	C0	Immediate	√	√	√	2^{19}	2^{10}
CPY *dp*	C4	Direct Page	√	√	√	2	$3^{2,10}$
CPY *addr*	CC	Absolute	√	√	√	3	4^{10}
DEC *Decrement* [Flags affected: n,z]							
DEC A	3A	Accumulator		√	√	1	2
DEC *dp*	C6	Direct Page	√	√	√	2	$5^{2,5}$
DEC *addr*	CE	Absolute	√	√	√	3	6^{5}
DEC *dp*,X	D6	DP Indexed,X	√	√	√	2	$6^{2,5}$
DEC *addr*,X	DE	Absolute Indexed,X	√	√	√	3	$7^{5,6}$
DEX *Decrement Index Register X* [Flags affected: n,z]							
DEX	CA	Implied	√	√	√	1	2
DEY *Decrement Index Register Y* [Flags affected: n,z]							
DEY	88	Implied	√	√	√	1	2
EOR *Exclusive-OR Accumulator with Memory* [Flags affected: n,z]							
EOR (*dp*,X)	41	DP Indexed Indirect,X	√	√	√	2	$6^{1,2}$
EOR *sr*,S	43	Stack Relative			√	2	4^{1}
EOR *dp*	45	Direct Page	√	√	√	2	$3^{1,2}$
EOR [*dp*]	47	DP Indirect Long			√	2	$6^{1,2}$
EOR #*const*	49	Immediate	√	√	√	2^{17}	2^{1}
EOR *addr*	4D	Absolute	√	√	√	3	4^{1}
EOR *long*	4F	Absolute Long			√	4	5^{1}
EOR (*dp*),Y	51	DP Indirect Indexed,Y	√	√	√	2	$5^{1,2,3}$
EOR (*dp*)	52	DP Indirect		√	√	2	$5^{1,2}$
EOR (*sr*,S),Y	53	SR Indirect Indexed,Y			√	2	7^{1}
EOR *dp*,X	55	DP Indexed,X	√	√	√	2	$4^{1,2}$
EOR [*dp*],Y	57	DP Indirect Long Indexed,Y			√	2	$6^{1,2}$
EOR *addr*,Y	59	Absolute Indexed,Y	√	√	√	3	$4^{1,3}$
EOR *addr*,X	5D	Absolute Indexed,X	√	√	√	3	$4^{1,3}$
EOR *long*,X	5F	Absolute Long Indexed,X			√	4	5^{1}
INC *Increment* [Flags affected: n,z]							
INC A	1A	Accumulator		√	√	1	2
INC *dp*	E6	Direct Page	√	√	√	2	$5^{2,5}$
INC *addr*	EE	Absolute	√	√	√	3	6^{5}
INC *dp*,X	F6	DP Indexed,X	√	√	√	2	$6^{2,5}$
INC *addr*,X	FE	Absolute Indexed,X	√	√	√	3	$7^{5,6}$

Assembler Example	HEX	Addressing Mode	02	C02	816	Bytes	Cycles
INX	*Increment Index Register X* [Flags affected: n,z]						
INX	E8	Implied	√	√	√	1	2
INY	*Increment Index Register Y* [Flags affected: n,z]						
INY	C8	Implied	√	√	√	1	2
JMP	*Jump* [Flags affected: none] [Alias: JML for all Long addressing modes]						
JMP *addr*	4C	Absolute	√	√	√	3	3
JMP *long*	5C	Absolute Long			√	4	4
JMP *(addr)*	6C	Absolute Indirect	√	√	√	3	$5^{11,12}$
JMP *(addr,X)*	7C	Absolute Indexed Indirect		√	√	3	6
JMP *[addr]*	DC	Absolute Indirect Long			√	3	6
JSR	*Jump to Subroutine* [Flags affected: none] [Alias: JSL for Absolute Long]						
JSR *addr*	20	Absolute	√	√	√	3	6
JSR *long*	22	Absolute Long			√	4	8
JSR *(addr,X)*	FC	Absolute Indexed Indirect			√	3	8
LDA	*Load Accumulator from Memory* [Flags affected: n,z]						
LDA (*dp*,X)	A1	DP Indexed Indirect,X	√	√	√	2	$6^{1,2}$
LDA *sr*,S	A3	Stack Relative			√	2	4^{1}
LDA *dp*	A5	Direct Page	√	√	√	2	$3^{1,2}$
LDA [*dp*]	A7	DP Indirect Long			√	2	$6^{1,2}$
LDA #*const*	A9	Immediate	√	√	√	2^{17}	2^{1}
LDA *addr*	AD	Absolute	√	√	√	3	4^{1}
LDA *long*	AF	Absolute Long			√	4	5^{1}
LDA (*dp*),Y	B1	DP Indirect Indexed,Y	√	√	√	2	$5^{1,2,3}$
LDA (*dp*)	B2	DP Indirect		√	√	2	$5^{1,2}$
LDA (*sr*,S),Y	B3	SR Indirect Indexed,Y			√	2	7^{1}
LDA *dp*,X	B5	DP Indexed,X	√	√	√	2	$4^{1,2}$
LDA [*dp*],Y	B7	DP Indirect Long Indexed,Y			√	2	$6^{1,2}$
LDA *addr*,Y	B9	Absolute Indexed,Y	√	√	√	3	$4^{1,3}$
LDA *addr*,X	BD	Absolute Indexed,X	√	√	√	3	$4^{1,3}$
LDA *long*,X	BF	Absolute Long Indexed,X			√	4	5^{1}
LDX	*Load Index Register X from Memory* [Flags affected: n,z]						
LDX #*const*	A2	Immediate	√	√	√	2^{19}	2^{10}
LDX *dp*	A6	Direct Page	√	√	√	2	$3^{2,10}$
LDX *addr*	AE	Absolute	√	√	√	3	4^{10}
LDX *dp*,Y	B6	DP Indexed,Y	√	√	√	2	$4^{2,10}$
LDX *addr*,Y	BE	Absolute Indexed,Y	√	√	√	3	$4^{3,10}$
LDY	*Load Index Register Y from Memory* [Flags affected: n,z]						
LDY #*const*	A0	Immediate	√	√	√	2^{19}	2^{10}
LDY *dp*	A4	Direct Page	√	√	√	2	$3^{2,10}$
LDY *addr*	AC	Absolute	√	√	√	3	4^{10}
LDY *dp*,X	B4	DP Indexed,X	√	√	√	2	$4^{2,10}$
LDY *addr*,X	BC	Absolute Indexed,X	√	√	√	3	$4^{3,10}$
LSR	*Logical Shift Memory or Accumulator Right* [Flags affected: n,z,c]						
LSR *dp*	46	Direct Page	√	√	√	2	$5^{2,5}$
LSR A	4A	Accumulator	√	√	√	1	2
LSR *addr*	4E	Absolute	√	√	√	3	6^{5}
LSR *dp*,X	56	DP Indexed,X	√	√	√	2	$6^{2,5}$
LSR *addr*,X	5E	Absolute Indexed,X	√	√	√	3	$7^{5,6}$
MVN	*Block Move Negative* [Flags affected: none] [Registers: X,Y,C]						
MVN *srcbk,destbk*	54	Block Move			√	3	1^{3}
MVP	*Block Move Positive* [Flags affected: none] [Registers: X,Y,C]						
MVN *srcbk,destbk*	44	Block Move			√	3	1^{3}
NOP	*No Operation* [Flags affected: none]						
NOP	EA	Implied	√	√	√	1	2
ORA	*OR Accumulator with Memory* [Flags affected: n,z]						
ORA (*dp*,X)	01	DP Indexed Indirect,X	√	√	√	2	$6^{1,2}$
ORA *sr*,S	03	Stack Relative			√	2	4^{1}
ORA *dp*	05	Direct Page	√	√	√	2	$3^{1,2}$
ORA [*dp*]	07	DP Indirect Long			√	2	$6^{1,2}$
ORA #*const*	09	Immediate	√	√	√	2^{17}	2^{1}
ORA *addr*	0D	Absolute	√	√	√	3	4^{1}
ORA *long*	0F	Absolute Long			√	4	5^{1}
ORA (*dp*),Y	11	DP Indirect Indexed,Y	√	√	√	2	$5^{1,2,3}$
ORA (*dp*)	12	DP Indirect		√	√	2	$5^{1,2}$
ORA (*sr*,S),Y	13	SR Indirect Indexed,Y			√	2	7^{1}
ORA *dp*,X	15	DP Indexed,X	√	√	√	2	$4^{1,2}$
ORA [*dp*],Y	17	DP Indirect Long Indexed,Y			√	2	$6^{1,2}$
ORA *addr*,Y	19	Absolute Indexed,Y	√	√	√	3	$4^{1,3}$
ORA *addr*,X	1D	Absolute Indexed,X	√	√	√	3	$4^{1,3}$
ORA *long*,X	1F	Absolute Long Indexed,X			√	4	5^{1}

Assembler Example	HEX	Addressing Mode	02	C02	816	Bytes	Cycles
PEA	*Push Effective Absolute Address* [Flags affected: none]						
PEA *addr*	F4	Stack (Absolute)			√	3	5
PEI	*Push Effective Indirect Address* [Flags affected: none]						
PEI (*dp*)	D4	Stack (DP Indirect)			√	2	6^{2}
PER	*Push Effective PC Relative Indirect Address* [Flags affected: none]						
PER *label*	62	Stack (PC Relative Long)			√	3	6
PHA	*Push Accumulator* [Flags affected: none]						
PHA	48	Stack (Push)	√	√	√	1	3^{1}
PHB	*Push Data Bank Register* [Flags affected: none]						
PHB	8B	Stack (Push)			√	1	3
PHD	*Push Direct Page Register* [Flags affected: none]						
PHD	0B	Stack (Push)			√	1	4
PHK	*Push Program Bank Register* [Flags affected: none]						
PHK	4B	Stack (Push)			√	1	3
PHP	*Push Processor Status Register* [Flags affected: none]						
PHP	08	Stack (Push)	√	√	√	1	3
PHX	*Push Index Register X* [Flags affected: none]						
PHX	DA	Stack (Push)		√	√	1	3^{10}
PHY	*Push Index Register Y* [Flags affected: none]						
PHY	5A	Stack (Push)		√	√	1	3^{10}
PLA	*Pull Accumulator* [Flags affected: n,z]						
PLA	68	Stack (Pull)	√	√	√	1	4^{1}
PLB	*Pull Data Bank Register* [Flags affected: n,z]						
PLB	AB	Stack (Pull)			√	1	4
PLD	*Pull Direct Page Register* [Flags affected: n,z]						
PLD	2B	Stack (Pull)			√	1	5
PLP	*Pull Processor Status Register* [Flags affected: n,z]						
PLP	28	Stack (Pull)	√	√	√	1	4
PLX	*Pull Index Register X* [Flags affected: n,z]						
PLX	FA	Stack (Pull)		√	√	1	4^{10}
PLY	*Pull Index Register Y* [Flags affected: n,z]						
PLY	7A	Stack (Pull)		√	√	1	4^{10}
REP	*Reset Processor Status Bits* [Flags affected: all except b per operand]						
REP #*const*	C2	Immediate			√	2	3
ROL	*Rotate Memory or Accumulator Left* [Flags affected: n,z,c]						
ROL *dp*	26	Direct Page	√	√	√	2	$5^{2,5}$
ROL A	2A	Accumulator	√	√	√	1	2
ROL *addr*	2E	Absolute	√	√	√	3	6^{5}
ROL *dp*,X	36	DP Indexed,X	√	√	√	2	$6^{2,5}$
ROL *addr*,X	3E	Absolute Indexed,X	√	√	√	3	$7^{5,6}$
ROR	*Rotate Memory or Accumulator Right* [Flags affected: n,z,c]						
ROR *dp*	66	Direct Page	√	√	√	2	$5^{2,5}$
ROR A	6A	Accumulator	√	√	√	1	2
ROR *addr*	6E	Absolute	√	√	√	3	6^{5}
ROR *dp*,X	76	DP Indexed,X	√	√	√	2	$6^{2,5}$
ROR *addr*,X	7E	Absolute Indexed,X	√	√	√	3	$7^{5,6}$
RTI	*Return from Interrupt* [Flags affected: all except b]						
RTI	40	Stack (RTI)	√	√	√	1	6^{9}
RTL	*Return from Subroutine Long* [Flags affected: none]						
RTL	6B	Stack (RTL)			√	1	6
RTS	*Return from Subroutine* [Flags affected: none]						
RTS	60	Stack (RTS)	√	√	√	1	6
SBC	*Subtract with Borrow from Accumulator* [Flags affected: n,v,z,c]						
SBC (*dp*,X)	E1	DP Indexed Indirect,X	√	√	√	2	$6^{1,2,4}$
SBC *sr*,S	E3	Stack Relative			√	2	$4^{1,4}$
SBC *dp*	E5	Direct Page	√	√	√	2	$3^{1,2,4}$
SBC [*dp*]	E7	DP Indirect Long			√	2	$6^{1,2,4}$
SBC #*const*	E9	Immediate	√	√	√	2^{17}	$2^{1,4}$
SBC *addr*	ED	Absolute	√	√	√	3	$4^{1,4}$
SBC *long*	EF	Absolute Long			√	4	$5^{1,4}$

Assembler Example	HEX	Addressing Mode	02	C02	816	Bytes	Cycles
SBC (*dp*),Y	F1	DP Indirect Indexed,Y	√	√	√	2	5[1,2,3,4]
SBC (*dp*)	F2	DP Indirect		√	√	2	5[1,2,4]
SBC (*sr*,S),Y	F3	SR Indirect Indexed,Y			√	2	7[1,4]
SBC *dp*,X	F5	DP Indexed,X	√	√	√	2	4[1,2,4]
SBC [*dp*],Y	F7	DP Indirect Long Indexed,Y			√	2	6[1,2,4]
SBC *addr*,Y	F9	Absolute Indexed,Y	√	√	√	3	4[1,3,4]
SBC *addr*,X	FD	Absolute Indexed,X	√	√	√	3	4[1,3,4]
SBC *long*,X	FF	Absolute Long Indexed,X			√	4	5[1,4]
SEC *Set Carry Flag* [Flags affected: c]							
SEC	38	Implied	√	√	√	1	2
SED *Set Decimal Flag* [Flags affected: d]							
SED	F8	Implied	√	√	√	1	2
SEI *Set Interrupt Disable Flag* [Flags affected: i]							
SEI	78	Implied	√	√	√	1	2
SEP *Set Processor Status Bits* [Flags affected: all except b per operand]							
SEP	E2	Immediate			√	2	3
STA *Store Accumulator to Memory* [Flags affected: none]							
STA (*dp*,X)	81	DP Indexed Indirect,X	√	√	√	2	6[1,2]
STA *sr*,S	83	Stack Relative			√	2	4[1]
STA *dp*	85	Direct Page	√	√	√	2	3[1,2]
STA [*dp*]	87	DP Indirect Long			√	2	6[1,2]
STA *addr*	8D	Absolute	√	√	√	3	4[1]
STA *long*	8F	Absolute Long			√	4	5[1]
STA (*dp*),Y	91	DP Indirect Indexed,Y	√	√	√	2	6[1,2]
STA (*dp*)	92	DP Indirect		√	√	2	5[1,2]
STA (*sr*,S),Y	93	SR Indirect Indexed,Y			√	2	7[1]
STA *dp*,X	95	DP Indexed,X	√	√	√	2	4[1,2]
STA [*dp*],Y	97	DP Indirect Long Indexed,Y			√	2	6[1,2]
STA *addr*,Y	99	Absolute Indexed,Y	√	√	√	3	5[1]
STA *addr*,X	9D	Absolute Indexed,X	√	√	√	3	5[1]
STA *long*,X	9F	Absolute Long Indexed,X			√	4	5[1]
STP *Stop Processor* [Flags affected: none]							
STP	DB	Implied			√	1	3[14]
STX *Store Index Register X to Memory* [Flags affected: none]							
STX *dp*	86	Direct Page	√	√	√	2	3[2,10]
STX *addr*	8E	Absolute	√	√	√	3	4[10]
STX *dp*,Y	96	DP Indexed,Y	√	√	√	2	4[2,10]
STY *Store Index Register Y to Memory* [Flags affected: none]							
STY *dp*	84	Direct Page	√	√	√	2	3[2,10]
STY *addr*	8C	Absolute	√	√	√	3	4[10]
STY *dp*,X	94	DP Indexed,X	√	√	√	2	4[2,10]
STZ *Store Zero to Memory* [Flags affected: none]							
STZ *dp*	64	Direct Page		√	√	2	3[1,2]
STZ *dp*,X	74	DP Indexed,X		√	√	2	4[1,2]
STZ *addr*	9C	Absolute		√	√	3	4[1]
STZ *addr*,X	9E	Absolute Indexed,X		√	√	3	5[1]

Assembler Example	HEX	Addressing Mode	02	C02	816	Bytes	Cycles
TAX *Transfer Accumulator to Index Register X* [Flags affected: n,z]							
TAX	AA	Implied	√	√	√	1	2
TAY *Transfer Accumulator to Index Register Y* [Flags affected: n,z]							
TAY	A8	Implied	√	√	√	1	2
TCD *Transfer 16-bit Accumulator to Direct Page Register* [Flags affected: n,z]							
TCD	5B	Implied			√	1	2
TCS *Transfer 16-bit Accumulator to Stack Pointer* [Flags affected: none]							
TCS	1B	Implied			√	1	2
TDC *Transfer Direct Page Register to 16-bit Accumulator* [Flags affected: n,z]							
TDC	7B	Implied			√	1	2
TRB *Test and Reset Memory Bits Against Accumulator* [Flags affected: z]							
TRB *dp*	14	Direct Page		√	√	2	5[2,5]
TRB *addr*	1C	Absolute		√	√	3	6[3]
TSB *Test and Set Memory Bits Against Accumulator* [Flags affected: z]							
TSB *dp*	04	Direct Page		√	√	2	5[2,5]
TSB *addr*	0C	Absolute		√	√	3	6[5]
TSC *Transfer Stack Pointer to 16-bit Accumulator* [Flags affected: n,z]							
TSC	3B	Implied			√	1	2
TSX *Transfer Stack Pointer to Index Register X* [Flags affected: n,z]							
TSX	BA	Implied	√	√	√	1	2
TXA *Transfer Index Register X to Accumulator* [Flags affected: n,z]							
TXA	8A	Implied	√	√	√	1	2
TXS *Transfer Index Register X to Stack Pointer* [Flags affected: none]							
TXS	9A	Implied	√	√	√	1	2
TXY *Transfer Index Register X to Index Register Y* [Flags affected: n,z]							
TXY	9B	Implied			√	1	2
TYA *Transfer Index Register Y to Accumulator* [Flags affected: n,z]							
TYA	98	Implied	√	√	√	1	2
TYX *Transfer Index Register Y to Index Register X* [Flags affected: n,z]							
TYX	BB	Implied			√	1	2
WAI *Wait for Interrupt* [Flags affected: none]							
WAI	CB	Implied			√	1	3[15]
WDM *Reserved for Future Expansion* [Flags affected: none (subject to change)]							
WDM	42	n/a			√	2[16]	n/a[16]
XBA *Exchange B and A 8-bit Accumulators* [Flags affected: n,z]							
XBA	EB	Implied			√	1	3
XCE *Exchange Carry and Emulation Flags* [Flags affected: m,b/x,c,e]							
XCE	FB	Implied			√	1	2

NOTES

[1]Add 1 cycie if m=0 (16-bit memory/accumulator)
[2]Add 1 cycle if low byte of Direct Page Register is non-zero
[3]Add 1 cycle if adding index crosses a page boundary
[4]Add 1 cycle if 65C02 and d=1 (65C02 in decimal mode)
[5]Add 2 cycles if m=0 (16-bit memory/accumulator)
[6]Subtract 1 cycle if 65C02 and no page boundary crossed
[7]Add 1 cycle if branch is taken
[8]Add 1 cycle if branch taken crosses page boundary on 6502, 65C02, or 65816's 6502 emulation mode (e=1)
[9]Add 1 cycle for 65816 native mode (e=0)
[10]Add 1 cycle if x=0 (16-bit index registers)
[11]Add 1 cycle if 65C02
[12]6502: Yields incorrect results if low byte of operand is $FF (i.e., operand is $xxFF)
[13]7 cycles per byte moved
[14]Uses 3 cycles to shut the processor down: additional cycles are required by reset to restart it
[15]Uses 3 cycles to shut the processor down: additional cycles are required by interrupt to restart it
[16]Byte and cycle counts subject to change in future processors which expand WDM into 2-byte opcode portions of instructions of varying lengths
[17]Add 1 byte if m=0 (16-bit memory/accumulator)
[18]Opcode is 1 byte, but program counter value pushed onto stack is incremented by 2 allowing for optional signature byte
[19]Add 1 byte if x=0 (16-bit index registers)

CMD's 2nd Annual

Warehouse Sale

Surplus Inventory Must Go by October 30, 1996

$2.00

< √	CBM 1660 (300 baud modem)
< #	Serial Cables 3'
† √	Midwest Micro 64 Terminal

$6.00

< √	BP Pro Joystick
! #	CBM Datasette (1530)
* #	California Games
+ #	C64 Graphics with COMAL
+ #	Commodore 64 Fun & Games
+ #	Commodore 64 Games
+ #	Computer Monsters
+ #	Computer Olympics
+ #	Computer Space Adventures
+ #	Computes 1st Book/C64
+ #	Computes 1st Book/C64 Games
+ #	Computes 2nd Book of C64
+ #	EPYX Programmers Basic Toolkit
+ #	Dictionary of Computer Terms
+ #	GEOS - Inside and Out
+ #	Instant Activities For Your C64
+ #	Jet: 82 New Adventures
+ #	Machine Language For Beginners
+ #	Mapping the Commodore 64
+ #	Micro Adventures #3 Million DollarGamble
+ #	Pilgrim in the Micro-World
+ #	Programming the Commodore 64
+ #	Write Your Own (Adv.Graphics Haunted House)
+ #	Write Your Own (Synthesizer)
* #	Alice in Wonderland
* #	Arcade Game Construction Kit
* #	Ardok the Barbarian
* #	Arkanoid
* #	Arkanoid II, Revenge of DOH
* #	Artic Fox
† #	Awardware
† #	Bank Street Filer
* #	Batman
* #	Batman - The Caped Crusader
* #	Beyond Castle Wolfenstein
* #	Bits, Pieces & Clues
* #	Blood Money
* #	Bubble Bobble
* #	Captain Commando Challenge
* #	Card Sharks
* #	Cauldron
* #	Cave of the Word Wizard
* #	Caveman UGH-lympics
* #	Championship Golf
* #	Chernobyl - Nuclear Plant Simul.
* √	Chomp
* #	Clowning Around
† #	Commodore Basic Programming
* #	Concentration
* √	Cut Throats
* #	Double Dragon II - The Revenge
* #	F-15 Strike Eagle
* #	Fairlight
* #	Family Feud
* #	Fast Tracks
* #	Feud
* #	Final Conflict
* #	Frankie Goes To Hollywood
* √	Frogger
* #	Fun House
* #	Gauntlet
* #	Ghostbusters
* √	Grand Prix Circuit
* √	Guerilla
* #	Hangman Roulette
* √	Hardball
* #	Heartland

$6.00

* #	Helicopter Emergency Rescue Operation
* √	High Roller
* #	Invaders of the Lost Tomb
* #	Jeopardy
* #	Jeopardy (Junior Edition)
† #	Jingle Disk - Holiday Card Maker
* √	Jordan vs. Bird
* #	Jumpman
* #	Karate Chop
* #	Ken Uston's Professional Blackjack
* #	King Cribbage
* √	Kings of the Beach
* #	Kung Fu II, Sticks of Death
† #	Label Maker
* √	Lords of Conquest
* #	Main Frame
* #	Maniac Mansion
* #	Math Blaster
* #	Maxwell Manor
* √	Mean Streets
* #	Realm of Impossibility
* #	Mini Putt
* √	Monday Night Football
* #	Monkey Business
* #	Murder By The Dozen
* #	Night Mission Pinball
* #	Operation Wolf
* #	PHM Pegasus
* #	Pinball Construction Kit
* #	Pole Position
* #	Powerplay Hockey - USA vs. USSR
* #	Predator
* #	Presidential Campaign
* #	Q-Bopper
* √	Qix
* #	Rack'em
* #	Rastan
* #	Remote Control
* #	Renegade
* #	Sargon II
* √	Sky Fox
* #	Snooper Troops Case#2
* √	Snow Strike
* #	Speedball
* #	Stationfall
* #	Summer Games II
* #	Super Huey
* #	Super Password
* #	Swiss Family Robinson
* #	Tag Team Wrestling
* #	Take Down
† #	The Artist
† #	The Home Manager
† #	The Money Manager
* √	The President Is Missing
* #	The Sentry
* #	The Seven Cities of Gold
* √	The Three Stooges V2
* #	The Wizard of Oz
† #	The Writer
* √	Thud Ridge
* √	Total Eclipse
* #	Trilogy
† #	Typing Tutor III
* √	War in Middle Earth
* #	Wheel of Fortune
* #	Wheel of Fortune 2nd Edition
* #	Where in the World is Carmen Sandiego
* #	Win, Lose or Draw 2nd Edition
† #	Word Writer
† #	Word Writer 3
* #	World Games
* #	Zenji

$6.00

• #	Atlantis
• #	Congo Bongo
• #	Gorf
• #	Omega Race
• #	Pirates Cove
• #	Practicalc 20
• #	Sargon II Chess
• #	The Sky Is Falling
• #	Tutankam
* #	Master Type 64 (Edu)
* #	Popeye
* #	Lazarian
* #	Omega Race

$8.00

+ √	Abacus - Anatomy of the 1541
+ √	Abacus - C128 Basic Training Guide
+ √	Abacus - C128 C.A.D.
+ √	Abacus - Ideas for use on C64
+ √	Abacus - Science & Engineering/64
+ √	Abacus - Tricks and Tips for C-64
† #	Data Manager 2
† #	Intro to Basic/ Part I
† #	Intro to Programming 1
† #	Music Construction Set
* √	Navy Seals
† #	Panorama
† #	Print Power
† #	Printed Word & Calc
† #	Programming Kit #3
† #	Swift Calc
† #	Word Writer 4
* #	Delta Drawing (Edu)

$10.00

< #	1541 Disk Drive (For Parts Only)
! #	Amtech (71) w/out PS (Parts Only)
< #	Epyx Fastload Cart.
! #	FSD-2 no P.S. (For Parts Only)
< #	M3 Mouse
< #	Mach 5 Cart.
< #	Navarone Port Expander
* √	AcroJet
* √	Airborn Ranger
* √	Atomino
* √	Ballistix
* √	Blood Money
* √	Dr.T's Midi Librarian (4 OP)
* √	Dr.T's Midi Librarian (CZ Rider)
* √	Dr.T's Midi Librarian (DX Heaven)
* #	Flight Simulator II Scenery Disk #1
* #	Flight Simulator II Scenery Disk #5
* √	Main Frame
* √	Menace
* #	Rings of Medusa
* √	Rings of Medusa
* √	Silent Service
* √	Skate or Die
* =	Ultima V (disks only)
• #	8k Ram Cartridge

$15.00

< √	1571 Drive Head Assembly
< #	1700 REU (128K)
< #	Aprotek Minimodem C (1200)
@ #	Cardco G-wiz
@ #	Datashare
Δ #	CadPak 128
* #	Flight Simulator II
Δ #	Home Designs (Cad for 128)
* #	Where in the USA is Carmen Sandiego
< #	Partner 64 Cartridge

Legend

(#) used (√) new (@) Printer Interface (◊) 64 Printer (*) Game (†) Productivity 64 (Δ) Productivity 128 (<) Misc. H'ware (!) Drive (%) Centronics Printer (•) Vic 20

$20.00

@ #	Cardco G+
@ #	Xetec Supergraphics, Jr.
Δ #	Data Manager 128
Δ #	Swift Calc 128
Δ #	Sylvia Porter's Personal Finance 128
Δ #	Word Writer 128
◊ #	Blue Chip / Protecto
@ #	Micro R&D MW-350

$25.00

% #	Star Gemini 10X
< #	Votalker (Speech Synthesizer)
< #	Xetec Super Graphics Jr.
† #	Accounting Series (Payables)
† #	Accounting Series (Receivable)
† #	Accounting Series (Invent. Cntrl)
† #	Accounting Series (Invoicing)

$30.00

◊ #	CBM 1526
◊ #	MPS-801
◊ #	MPS-802
◊ #	MPS-803
◊ #	Seikosha SP-1000VC
< #	Animation Station & Design Pad
< #	Voice Messenger w/Easy Speech & Intelligent Talking Terminal

$40.00

Δ #	Vizastar 128 (SS,dB, Graphics)
Δ #	Vizawrite 128 (WP)
! #	Amtech RF512C (71 clone)w/ PS
! #	Enhancer 2000 w/ PS(41 Clone)
% #	Epson FX-85 $40.00
◊ #	Okimate 10 $40.00
◊ #	Okimate 20 $40.00
% #	Okimate 20 (Centronics)
% #	Laser 190E (Epson 9 Pin)
◊ #	MPS-1000 w/out Tractor

Price as Marked

< √	50 - 5.25" DSDD Disks in Storage Case **$20.00**
< √	25 - 5.25" DSDD Disks in Storage Case **$12.00**
< √	1670 modem (1200 baud) **$14.00**
! #	FSD-2 Drive (1541) w/ PS **$50.00**
◊ #	MPS-1000 w/ Tractor **$55.00**
◊ #	Okidata 120 **$60.00**
< #	1701/1702 Color Monitor **$79.00**
◊ #	Star NX-1000C **$75.00**
< √	RAMLink 1MB (Rev.1) **$149.00**
< √	CMD HD-100 Hard Drive **$239.00**

Ordering Details

Minimum total order $20.00. No product will be held awaiting check/MO. We suggest payment by major Credit Card (MC/Visa/Disc./Amex).

All items have been tested and are guarranteed to work upon delivery. CMD will warranty above merchandise for 10 days. Defects may be returned for replacement or merchandise credit only with prior authorization. No Refunds.

Shipping Cont. US only:Printers $10.00; Drives $8.50; Monitors $20.00. All other products will be shipped based on the shipping chart found in our main ad elsewhere in this magazine.

CMD • HARDWARE • ORDERS: 1-800-638-3263 • SOFTWARE • CMD

HD Series SCSI Hard Disk Drives

HD-40 (40 MB)	**$249.00**
HD-170, 170 MB (Special Edition)	**$299.00**
HD-500, 500+ MB (Special Edition)	**$379.00**
HD-1000, 1 GB (Special Edition)	**CALL**
HD-2000, 2 GB (Special Edition)	**$659.00**

FD Series 3.5" Floppy Disk Drives

FD-2000 (800K and 1.6 MB)	**$159.00**
FD Real-Time-Clock Option/Kit	$20.00/$29.00
Box of 10, High Density Disks (1.6MB)	$14.95
Box of 10, Enhanced Density Disks (3.2 MB)	$29.00

JiffyDOS

(Specify computer serial number and drive model)

JiffyDOS C64/SX-64 System	**$39.00**
JiffyDOS C-128/128-D System	**$49.00**
Additional Drive ROM's	**$19.00**

Miscellaneous Hardware

80-column Monitors (Refurb)	CALL
Action-Replay Cartridge	$59.95
Aprotek 3-Way User Port Expander	$30.00
Aprotek User Port Extension Cable	$19.00
C-64 Computer (Refurb, w/JD)	$99.00
C-64C Computer (Refurb, w/JD)	$119.00
C-64/64C Power Supply (Repairable)	$39.00
C-64/64C Heavy Duty Power Supply (Repairable)	$56.95
128-D Computer w/ Int. 1571 (Refurb, w/JD)	CALL
C-128 Computer (Refurb, w/JD)	$179.00
C-128 Computer (Refurb, w/JD, no PS)	$129.00
C-128 Power Supply (Repairable)	$56.95
Cannon BJC-4100 Color Bubble Jet Printer	$349.00
Commodore 1541 Disk Drives (Refurb, w/JD)	$89.00
Commodore 1541-II Disk Drive (Refurb, w/JD)	$109.00
Commodore 1571 Disk Drive (Refurb, w/ JD)	$119.00
Commodore 1702 40-column Monitor (Refurb)	$79.00
Commodore 1802 40-column Monitor (Refurb)	$139.00
CMD EX2+1 3-Port Cartridge Port Expander	$34.95
CMD EX3 3-Port Cartridge Port Expander	$29.95
CMD Gamepad/Joystick Controller	$24.95
CMD GeoCable II Print Cable 6ft./15ft.	$29.00/$34.00
CMD SmartMouse (1351 Compatible Mouse)	$49.95
CMD SmartTrack (1351 Compatible Trackball)	$69.95
Inkwell Light Pen Model 170C	$75.00
Monitor Cables	CALL
Mouse Pad	$2.95
MW-350 Printer Interface (ØK/8K Buffer)	$49.00/$60.00
Panasonic KXP-1150 9-pin Printer	$169.00
Panasonic KXP-2023 24-pin Printer	$229.00
Panasonic KXP-2135 24-pin Color Printer	$319.00

Utilities

Big Blue Reader V4.10 (SOGWAP)	$39.00
CMD Utilities	$24.95
JiffyMON-64 (ML Monitor)	$19.95
The Compression Kit '94 (Mad Man)	$39.00

GEOS

Collette Utilities (Handy Geos Utilities)	$19.95
Desk Pack Plus	$29.00
Dweezils Greatest Hits (NewTools2,Stamp,Label128)	$30.00
FONTPACK Plus	$25.00
gateWay 64 or 128 (Specify Version)	$29.95
geoBASIC	$20.00
geoCalc 64/128	$40.00/$45.00
geoChart	$29.00
geoFAX	$39.95
geoFile 64/128	$40.00/$45.00
geoMakeBoot (Makes Bootable copies)	$12.95
geoProgrammer	$45.00
geoPublish	$40.00
GEOS 64 v2.0	$44.00
GEOS 128 v2.0	$49.00
geoSHELL V2.2 (CLI for GEOS)	$24.95
International FONTPACK	$25.00
Perfect Print LQ for GEOS (Laser-like output)	$49.95
RUN GEOS Companion	$20.00
RUN GEOS Power Pak I or II (Specify)	$20.00

Books

Anatomy of the 1541	$8.00
C-64 Science & Engineering	$8.00
C128 Computer Aided Design	$8.00
C128 BASIC Training Guide	$8.00
Commodore 64 Tricks and Tips	$8.00
GEOS Programmers Reference Guide	$8.00
Graphics Book for the C-64	$8.00
Hitchhikers Guide to GEOS	$8.00
Ideas for Use on Your C-64	$8.00
Printer Book for the C64	$8.00
Mapping the C64	$8.00
Simple Internet	SOLD OUT!

Scanning & Video

Handyscanner	$249.00
Pagefox	$139.00
Video Digitizer	$189.00
Videofox	$89.00

SuperCPU-64 Accelerator $199.00

RAMLink Power-Backed RAM Disk

RAMLink Base Model (ØMB, No RAMCard)	$149.00
RAMLink w/1 MB RAMCard (Special!)	**$189.00**
RAMLink w/4 MB RAMCard (Special!)	**$229.00**
RAMCard RTC Option/Kit	$10.00/$19.00
RAMLink Battery Back-up (Optional)	$24.95
Parallel Cable (RAMLink to HD)	$14.95

MIDI & Sound

Digimaster	$34.95
SID Symphony Stereo Cartridge	$44.95
Sonus 64 Sequencer, MIDI Interface, Cables	$99.00
Sonus 128 Sequencer, MIDI Interface, Cables	$99.00

Languages & Compilers

BASIC 64 Compiler (Abacus)	$17.00
BASIC 128 Compiler (Abacus)	$25.00
Blitz! 64 Compiler (Skyles)	$30.00
Blitz! 128 Compiler (Skyles)	$30.00
Buddy 64/128 Assembler	$39.00
Cobol 64 (Abacus)	$17.00
Fortran 64 (Abacus)	$17.00
Pascal 64 (Abacus)	$17.00
Power C 64 (Spinnaker)	$14.00

Productivity

Cadpak 64 (Abacus)	$22.00
Cadpak 128 (Abacus)	$25.00
Chartpak 64 (Abacus)	$17.00
Chartpak 128 (Abacus)	$25.00
Data Manager 64 (Timeworks)	$16.00
I Paint v1.5 (128, 80-col., 64K VDC) (Living Proof)	$39.00
I Port v1.54 (128, 80-col., 64K VDC) (Living Proof)	$29.00
Personal Portfolio Manager (Abacus)	$16.00
Pocket Writer 1 (64) Digital Sol.)	$35.00
Pocket Writer 2 (64/128) (Digital Sol.)	$65.00
Pocket Writer 3 (64 or 128) (Digital Sol.)	$70.00
Pocket Planner 2 or Pocket Filer 2 (Digital Sol.)	$35.00
RUN Productivity Pak I, II, or III (Specify)	$15.00
RUN Super Starter Pak 1541 or 1581	$20.00
RUN Works	$20.00
SEC Check Register 128	$50.00
SuperScript 64 (Precision)	$15.00
SuperScript 128 (Precision)	$20.00
Suberbase 64 Version 3.01 (Precision)	$35.00
Superbase 128 Version 3.01 (Precision)	$35.00
SwiftCalc 64 (Timeworks)	$16.00
TWS 64 w/Speller (Busy Bee)	$29.00
TWS 128 w/Speller (Busy Bee)	$39.00
TWS Modules (HD/RL/Illustrator)	each $5.00

Telecommunications

Aprotek MiniModem C-24 (C= ready, 2400 baud)	$69.00
Aprotek MiniModem C (C= ready, 1200 baud)	$50.00
Aprotek Com-Modem Adapter (for ext. modem)	$19.00
Aprotek Com-Modem Adapter Cable	$9.95
BOCA 2400 w/SwiftLink and Cable	$99.00
BOCA 14.4K bps FaxModem	$129.00
BOCA 14.4K w/SwiftLink & Cable	$159.00
BOCA V.34 28.8K bps FaxModem	$199.00
BOCA V.34 w/SwiftLink & Cable	$229.00
Commodore 1670 Modem (1200 baud)	$14.95
Dialogue 128	$29.00
Nova Term Version 9.6 ON DISK!!!	$29.95
SwiftLink RS-232 Cartridge (Up to 38.4K baud)	$39.95
SwiftLink Modem Cable (DB9-DB25)	$9.95

Games

Atomino	$17.00
Ballistix	$13.00
Blood Money	$15.00
Chomp!	$10.00
Day in the Life of Prehistoric Man	$19.00
Escape Route	$19.00
Flummi's World	$19.95
Heavenbound	$19.95
Island of the Dragon	$19.00
Laser Squad	$13.00
Lazer Duel	$14.95
Lions of the Universe	$19.95
Mainframe	$13.00
Menace	$15.00
Navy Seal	$10.00
Rings of Medusa	$16.00
RUN C128 Funpak	$10.00
RUN C64 Gamepak	$10.00
Shoot 'Em Up Pack (Gangster/Time Traveler)	$14.95
Skate or Die	$13.00
Slaterman	$14.95
The Amazing Spider-Man	$15.00
The President Is Missing!	$10.00
The Three Stooges	$10.00
Tie Break Tennis	$16.00
Total Eclipse	$10.00
Walkerz	$19.00
Wings of Circe	$19.00

Novaterm

version 9.6

NEW

Join the Telecommunications Revolution... get on the Information Super Highway TODAY!

- **Easy-to-use — novice user configuration**
- Now supports **UUencode and UUdecode** and converts files on disk or in a buffer. Also ASCII <> PETSCII and PETSCII to Unix.
- Protocal support: **Zmodem - up/down load, resume (crash recovery), streaming to buffer**; Ymodem batch, **Ymodem-g**; Xmodem-1k, **Xmodem-1k-g**(to buffer) Xmodem-CRC; Punter; Kermit; WXmodem.
- Supports High Speed Interfaces like SwiftLink for baud rates up to 38,400 baud
- Supports ANSI color/graphics, VT102, and VT52 in 80 col. Plus C= color graphics and ANSI color in 40 col. **All emulation modules now load from one window.**
- New faster 80 column modes: **Soft-80 on C-64 (enhanced scrolling w/ REU); and 25- or 28-line C-128 VDC 80-col. mode.**
- Now **use any memory device as a buffer** including: 17xx REU, RAMLink partition, GeoRAM, C128 VDC, BBGRAM, and internal C64 memory.
- New text capture features:Capture text on-line and store in any device; High-speed transfers directly into memory and now **retain buffer contents with battery/power backed RAM devices such as RAMLink.**
- Enhanced full featured **Text editor can now load/save files from the buffer** and has an integrated script compiler.
- Script language for automatic operations
- Simple BBS mode - dial-in downloading
- Supports **real-time clocks in CMD devices**
- **All new user's manual**

Now Just $29.95

(Specify 5.25" or 3.5" disk)

JiffyDOS

Increase Speed Up to 1500% while retaining 100% compatibility

64 System Computer & Drive ROM $39.00

Add'l Drive ROMs $19.00

128 system Computer & Drive ROM $49.00

Note: Computer Serial Number Required for C-64/64C Orders

COMMODORE WORLD

THE NEWS MAGAZINE FOR C-64 & 128 USERS

For more than two years, Commodore World has been keeping the C-64/128 market alive with up-to-date information from some of the brightest minds in the Commodore community. From beginner to expert, CW has something for everyone. Columns regularly cover: Programming, what's new, reviews, games, GEOS, telecom, projects and much more! Stop listening to everyone telling you what you can't do with a Commodore and see what the latest innovations can do for you!

Subscribe for only $29.95

Canada $35.95, EC Countries $45.95, Other Foreign $57.95

Shipping and Handling Charges

	Continental United States					
Order Subtotal	UPS Ground	2nd Day	Next Day	AK HI, PR	Canada	Foreign
$0.01 to $19.99	$4.50	$13.00	CALL	$15.00	$5.00	CALL
$20.00 to $29.99	$5.50	$17.00	CALL	$17.00	$7.00	CALL
$30.00 to $59.99	$6.50	$19.00	CALL	$19.00	$9.00	CALL
$60.00 to $149.99	$7.50	$21.00	CALL	$28.00	$12.00	CALL
$150.00 to $299.99	$10.00	$25.00	CALL	$35.00	$20.00	CALL
$300.00 to $799.99	$15.00	$29.00	CALL	$40.00	$25.00	CALL
$800.00 and up	$20.00	CALL	CALL	$50.00	$38.50	CALL

UPS C.O.D. add $5.00 (US/PR only)

Policies: CMD accepts: Major Credit Cards, Money Orders, COD and Personal Checks. Personal Checks are held 3 weeks. Customer is responsible for shipping charges on refusals. Most items are stock, contact CMD for firm delivery. Returns for merchandise credit only within 30 days with prior authorization. Credits are less shipping, handling, taxes, customs, duties, and a 10% restocking charge. No refunds or credits on opened software. All prices and specifications are subject to change without notice.

••• Major Credit Cards Accepted •••

Mastercard • Visa • Discover • American Express

Creative Micro Designs, Inc.

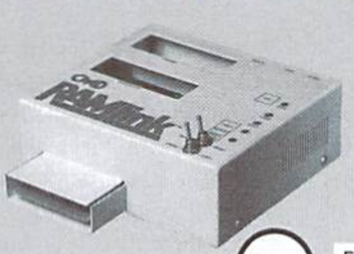
P.O. Box 646
East Longmeadow, MA 01028

Info: (413) 525-0023
Fax: (413) 525-0147

Special Pricing

On the fastest, most advanced storage devices available for the C-64 and C-128 computer

RAMLink™



Power-Backed Expandable RAM-disk Drive

- Fastest possible method of storage with capacities up to 16 MB. Power and battery back-up retains data even when computer is turned off!
- Equipped with CMD DOS for flexibility and optimum compatibility, RAMLink can emulate multiple 1541, 1571, & 1581 disks or use CMD Native partitions.
- Ideal for use with GEOS as RAM Disk, RAM Expansion or both.
- Easily connects to cartridge/expansion port yet behaves and operates like a super-fast disk drive with built-in JiffyDOS and SWAP features.
- Built-in Pass-thru port for other cartridges, RAM-Port for REU's & GEORAM and Parallel port for fast access to CMD hard drives.

FASTEST

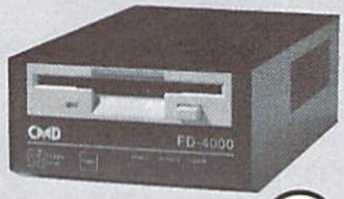
RL w/ 1 MB $189⁰⁰ • RL w/ 4 MB $229⁰⁰

*Add Battery & RTC $25⁰⁰**

Full Featured High Density 3.5" Disk Drive

FD Series™ Disk Drives

- The FD-2000 uses modern High Density diskettes (1.6 MB) just like Pentiums or the Power PCs. The FD-2000 is the highest capacity floppy drive available for the C-64/128 and is more compatible with existing software than the 1581!
- Equipped with CMD DOS for compatibility and speed, the FD-2000 also reads and writes 1581 disks, and can emulate multiple 1541 or 1571 disks.
- Easily connects to the serial bus just like any other drive and features JiffyDOS for greatly enhanced speed on JiffyDOS equipped computers.
- Features: SWAP 8/9 switch, Device # switches (8-15), GEOS compatibility software, FD Utilities disk and very small size (7.625"L x 4.75" W x 2.375"H).



FAST

FD-2000 Only $159⁰⁰

Real-Time Clock add $20⁰⁰

HD Series™

Fast SCSI Hard Disk Drives for the C-64/128

HD Series Hard Drives are available in capacities up to 2 GB, are fully partitionable, and can emulate 1541, 1571, & 1581 disks as well as provide Native partitions which can be up to 16 MB in size and utilize MS-DOS-style subdirectories. HD's connect easily to the serial bus or to RAMLink via a high-speed parallel cable. They are many times faster than any other type of physical disk drive. Includes built-in JiffyDOS, SWAP feature and Real-Time Clock. HD's offer superior compatibility with most commercial software including BBS, Productivity and GEOS. And with new pricing, HD Series drives offer the lowest cost per megabyte of any C64/128 storage device. Imagine... No more thumbing through endless piles of disks looking for the right one — use your hard drive like a huge file cabinet to organize and store hundreds or thousands of disks. Each 40 MB holds up to 240 -1541 disks. Get yours Today!

FASTER

40 MB Only $249⁰⁰

170 MB	$299⁰⁰	•	500 MB	$389⁰⁰
1,000 MB	CALL	•	2000 MB	$659⁰⁰

* Only valid at time of RAMLink purchase.

Offers valid only in U.S. and Canada through October 31, 1996. Prices do not include shipping and handling. Pricing and spec's subject to change without notice. See our main ad for shipping prices and policies. May not be combined with any other offers.

GAMES

Forgotten Worlds

Capcom (available from SSI)

Heroes have it rough. They're constantly being outfitted with the barest essentials and shoved out to take on a slew of hostile aliens with little or no hope of survival, with the small comforts of the occasional weapons shop to keep them company.

That's Forgotten Worlds for you. Capcom wastes no time (the manual is barely two pages long) setting the stage for you and hurling you into the action. You, the hero, outfitted with an anti-gravity device and a photon gun, must take on Lord Bios' swarm of alien invaders who threaten to conquer the Earth. Nobody said life would be easy...

You progress across a scrolling landscape as wave after wave of nicely detailed opponent comes at you. Your photon gun can fire in eight directions as you fly across the screen, which is a good thing, because the enemy comes from all angles. Successful kills often leave behind Zennys—coins you can collect for later use toward improved weaponry at the strategically located weaponry stores.

The action is fast-paced and often the enemy seems to be endless, so you may want to bring a friend who plays simultaneously. This allows for some much-needed strategy. For example, one player can cover ground enemies while the other picks off airborne targets, or one player can take on enemies that approach from the right of the screen (in the direction of motion) while the other covers those that attack from behind. Either way, there will be plenty to do.

You, the hero, are imbued with a great deal of energy, which is depleted by enemy fire and collisions. This energy supply can be recharged at shops, but that can get expensive quickly. You have but one life, although you have a limited number of continuation credits at your disposal.

The graphics are amazingly rich in this game, from the loading title screen onward. Unfortunately, the programmers did not cover all of their tracks and you may find the top half of your character disappears if you venture too near the top of the screen while there is a lot of action going on. This can get a little distracting when the fleet is almost upon you.

I can't stress enough how much better this game is with two players. The ability to aim your gun is a great one, but the control system makes it somewhat difficult to keep your gun pointed in one direction while you move in another. Having a backup increases your odds of surviving. Even when you master the firing angles, the enemy comes from so many places it's good to have someone to watch your back.

On the single-player mode, consider yourself a skilled beginner if you can clear the first few waves of attackers without losing all of your energy. The attacks are that intense. After a while, you start to settle into a rhythm, which gets shaken up as soon as the attacks start in from other angles.

Having another player around is a great boon in this case, and it's not many action games that allow for this sort of cooperation, so it would be best to take advantage of it while you can.

Forgotten Worlds does a great job of creating a graphically atmospheric realm for you to do battle on, and has an excellent two-player option. It is a shame that the follow-through on the more ambitious visual effects was not very strong, and of course the control system does leave something to be desired. But Forgotten Worlds is a great space-blasting adventure in its own right.

-Jason Compton

Grand Prix Circuit

Accolade (available from SSI)

Have you ever climbed behind the wheel of your car and wished you were climbing into a Formula One race car? Have you ever stopped at a red light and revved your engine for that ultimate moment when it finally turns green? Do you get a thrill from being the first one off the block at that same red light? Do you love the challenge of a narrow winding road? Well, if you've answered yes to any or all of the above questions, you are ready for the Challenge of the Grand Prix Circuit.

Accolade presents the Commodore owner with the challenge of a lifetime, that of the Grand Prix Circuit, a Formula One Racing Simulation. For those of you not familiar with the Grand Prix Circuit, it is a racing tour open only to Formula One race cars. The races themselves are held on specially constructed tracks filled with challenging twists, turns, and even tunnels. The Grand Prix has strict qualifying requirements and presently there are only around 30 drivers in the entire world who have what it takes to enter this Formula One event. So as you can see, it's a challenge few have managed to conquer. Keep this in mind as you settle down into your favorite computer chair to race in the Grand Prix Circuit.

When the game begins you are first presented with an option screen. From here you can choose to practice, partake in a single race or the championship circuit. Below these options is a difficulty level bar. There are five different levels of difficulty. The first two levels are for beginners and provide you with automatic shifting and the inability to blow an engine. Level three is the first real driving level. Here you get to shift your gears yourself and your car begins to get damaged much more easily. In level four you begin to realize that you're not racing against your buddies but real opponents who want to win as much as you do. And finally, in level five, you meet up with the pros of the Grand Prix Circuit where anything can happen. Next you'll need to type in your name and choose the number of laps you'd like for each race. I love having the freedom to choose how many laps I want to race because this means I have the choice of playing a long or short game.

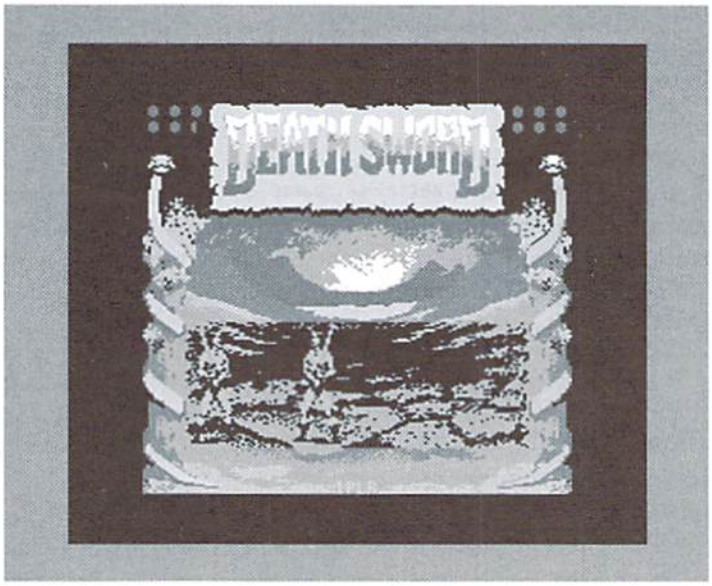
After you've completed these options, you're whisked away to the next screen where you get to view each of the eight different tracks of the Circuit. The tracks are located in Brazil, Monaco, Canada, Detroit, Britain, Germany, Italy, and Japan. For the first time player, I recommend choosing the Hockenheim-Ring in Germany. Although it presents you with a few amazing bends, it's certainly a lot easier than the loops and tunnels of the Suzuka International Race Course in Japan.

Finally, it's time to choose your car. There are three Formula One race cars to choose from: the Ferrari—the best of the three for the beginning player because it's the easiest to steer; the Williams—good for the intermediate player because of its speed and braking abilities; and the McLaren—the speediest of the three and the most difficult to handle. The game displays each of the three cars on a screen of its own along with its relative statistics and features. Once you've made your decision, it's off to the races you go!

Well now, how does the game play? As I sat in my chair I found myself actually twisting and turning with the bends. It's pretty realistic. The graphics are gorgeous, giving you the feeling of actually being in all of the eight different countries and the sound and music are just as good. But, the game is difficult. Even in the practice levels I found myself ruining my car for quite some time. What I finally realized is that speed isn't everything in the Grand Prix Circuit. Sometimes due to the all the bends you'll find that you just have to take it slow. Once you've realized this, you'll find yourself actually finishing the races and then it's finally time to try the single race. The real races outside of the practice races, require you to qualify before you actually partake in a race. I had no problem qualifying. I even managed to qualify for the ninth position! And just as surprising I managed to complete my first actual race and be in the top five! Of course this was at the easiest difficulty setting.

To rap it up, Grand Prix is what you would expect from a racing simulation. However, because it is the Grand Prix you'll find it more challenging than other games of this variety. Who knows, perhaps you could be the next Mario Andretti!!

-Sherry Freedline



Death Sword

Epyx (available from SSI)

Death Sword is a one- or two-player sword fight simulation for the Commodore 64. A joystick is required for each player.

In Death Sword you take on the role of Gorth, a warrior prince of the Northlands. During your travels you learn of Drax, an evil sorcerer, who has seized the city's throne and, even worse yet, imprisoned their princess with the intentions of making her his future bride. Of course, being the noble, fearless warrior that you are, you decide to take on the challenge of rescuing the princess and restoring her to her rightful throne. By doing so, you will become a legendary hero and be awarded wealth and riches befitting such an accomplishment.

On side one of the Death Sword disk is a practice game. Here your companions will help you train for your battles. When you feel satisfied that you've become an accomplished swordsman, flip the disk over to begin your search for Drax.

Before you can fight Drax for the princess' freedom, you must fight and overcome the warriors located in the castle's throne room, the enchanted forest, the lava pit, and then in the palace's dungeon. If you lose, the sorcerer's pet, Grundel—a slobbering green creature—will have you for lunch!

All in all it sounds like a fun game. After reading the manual I was quite eager to play the game. I plopped the game in the disk drive, loaded it up, went to grab a drink and came back to find a major disappointment. Death Sword is definitely not a game of commercial quality. The plot, the goal, and even the game play are all what you would expect to find in a good commercial game. However, it's the graphics and sound that cause the feeling of disappointment. Aside from the clinking and clanging of the swords and an occasional small musical sound effect, the game is silent.

Fighting a warrior is done through the use of your joystick. For instance by moving the joystick upward you can jump, move it in the southwest position to roll backward. Pressing the joystick button while moving the joystick gives you additional moves such as overhead chops, head-butts, and the web of death. The practice mode comes in handy for learning all these combinations. Despite the low quality graphics and sound, I did still enjoy the fight.

As mentioned above, the game can be played by one or two players. In the one player game there are no time restrictions involved when it comes to fighting the enemy. However, in the two player games the fights can last up to a total of ninety seconds. If you and your friend survive this time period the battle is considered won and you will regain your strength in preparation for the next battle.

So, I guess when it comes down to the option of purchasing or not purchasing this game you'll have to weigh your odds. If you like games of this nature involving a good fight, you will probably be able to overlook the weaknesses in the graphics and sound. However, if you're simply looking for a diversion you may want to look elsewhere.

-Sherry Freedline

BOOKWARE IN REVIEW

The Software Manual

Programming the 65816

David Eyes and Ron Lichty; $65.00; The Western Design Center, Inc., 2166 East Brown Road, Mesa, AZ 85213; (602) 962-4545

All of the discussion surrounding the CMD SuperCPU introduction has raised questions about the 65C816S CPU that forms the heart of the CMD unit and what advantages it offers over the 6502 CPU. For those interested in learning the specifics on this new IC, help is available.

Western Design Center, manufacturer's of the CPU, offers a book designed to help answer questions concerning the WDC line of CPUs and how to develop applications on them. Titled "Programming the 65816, Including the 6502, 65C02, and 65802" is written by David Eyes and Ron Lichty, and is offered by WDC as a reference manual. Although written in 1985, the information is timely and accurate.

The book is divided into 4 main sections and contains a number of appendices. Assuming nothing about the reader's prior knowledge of the 65XX line of CPUs or programming in assembly language, the first section details concepts crucial to understanding CPU operation and software development.

Topics covered include:

- Descriptions of bits, nybbles, and bytes
- Displaying quantities in hexadecimal format
- Boolean Logic
- Performing arithmetic in a CPU
- Writing programs in assembly language

With the reader having some grasp of the basics, section 2 details the architecture of the three main microprocessors described in this manual, the 6502, the 65C02, and the 65816. Many advanced readers will find the 6502 chapter of only slight importance, but inexperienced programmers will find the chapter full of important insights into the workings of the 65XX CPU line. The basic register set and status flags are described, as are the various instructions and the layout of memory. Stack operations and the stack page are also detailed. At the close of the chapter, various "bug" or quirks present in the original 6502 as designed by MOS Technologies. Most are completely accurate, although this reference, as well as the various WDC data sheets, specifies that the NMOS decimal mode leaves important registers in unknown states. Although this might be true, information to the contrary has been presented by Commodore users.

Chapter 3 details the 65C02, the CMOS 6502 designed by WDC to replace the original 6502. This chapter is very brief, describing only the 27 new opcodes available in this variant of the CPU. For advanced developers, the 65C02 brings with it the ability to branch unconditionally (BRA), as well as the ability to push and pull the index registers off the stack. (No more PLA, TAY, PLA, TAX). One section describes the advantages of the CMOS fabrication process over the NMOS process used on the 65XX CPUs used in the Commodore line of computers. Finally, the last paragraph describes the differences between the NMOS 6502 and the CMOS 65C02 CPUs. Programmers will be happy to note that the quirks of the NMOS 6502 have been fixed. However, in some ways, the 65C02 is too good. In fixing the problems in the NMOS 6502, some opcodes in the 65C02 actually execute faster. In a few instances, the extra speed might wreak some havoc, but the ending sentence in Chapter 3 claims that "this has affected little software".

For programmers eager to start developing SuperCPU native mode applications and take full advantage of the powerful 65816 should pay special attention to this book starting with Chapter 4. This chapter introduces the 65816 and the 65802 (the version of the 65816 that is pin compatible with the 6502/C02). The two "modes" (emulation and native) are discussed, as is which mode the CPU powers up in (emulation). The 65802's restrictions compared to the 65816 are described in passing, and the important advantages of the 65816 and 65802 are detailed. Many already know the 65816 can address up to 16 megabytes of memory directly, but few will note that stack and zero page have been widened to include all of the first 64 kilobyte bank of memory.

Detailed diagrams of the 65816 register set are illustrated, and some nomenclature specific to the 65816 is discussed in this chapter. The book shows programmers how to view the 16 megabytes in "banks", as well as "pages". Direct Page (the expansion of zero page) is described in some detail, and each new register is outlined with its functions detailed. Introduced in the chapter as well is the notion of 16 bit accumulators and index registers. As this concept and how to utilize the resulting registers subtly affects programming choices, the books takes ample time to discuss this new feature. As programmers might guess, the expansion of these registers affect standard 6502 opcodes, and the books describes this as well. The 9 new addressing modes available in the 65816 are outlined, as are the 78 new opcodes available in this CPU. The remainder of this chapter describes the 65802 and how it differs from the 65816. Interestingly, this chapter closes describing how the 65816 and 65802 provide slightly better compatibility with the 6502 than the 65C02 does. The cause: The 65816/802 does not alter any timing of any 6502 opcodes, whereas the 65C02 does.

Part 3 of this book begins the Tutorial Section. This section begins by outlining the REP and SEP opcodes, used to switch the 65816 between modes. Then, the assembler notation used throughout the tutorial section is described. Readers should beware that, although the concepts and codes presented in this tutorial are accurate, the assembler syntax and some directives used in the samples will not be familiar to Commodore developers. The samples were developed on an Apple computer using an Apple assembler, so some alterations by the reader will need to be made in order to create correctly assembled samples. To give some indication of this sections importance, it occupies almost 200 pages, while the previous 2 receive only 75 pages.

Much of this tutorial information applies to all the CPU models referenced by the book's title. However, some information and some addressing modes are only applicable to the 65816 and 65802. Like all machine language tutorials, this one shows how to move data around, how to use the various addressing modes on each CPU, how to alter the flow of a program, and how to perform simple math using the CPU. Further chapters describe logic and bit operations, utilizing the complex addressing modes available on the 65816/802, how to write programs as collections of "subroutines", and how to take advantage of "interrupts". Veteran developers will find much of the information redundant, but new programmers may find themselves overwhelmed. I would suggest attempting the tutorial in stages.

Section 4 provides some examples of code used to perform real world work, from multiplying numbers to dividing them. An entire chapter is devoted to describing and implementing a 65816 step and trace debugger called DEBUG16. A final chapter in this section outlines several "gotchas" in developing applications and offers ways to alleviate or eliminate these problems.

If, perchance, you are the type who can quote hex in your sleep, can assemble complex programs by hand or in your head in minutes, dream in binary, or is referred to by others in hushed tones as "guru" or "master programmer", you can safely skip to the 5th and final section. This section provides a wealth of reference materials for all. Chapter 17 describes each of the 65XX/65XXX addressing modes in detail, while Chapter 18 describes each instruction available in this line of CPUs. The final chapter, 19, lists all the instructions and which CPU can utilize each one. In addition, the complete opcode matrix table showing opcode values, instruction lengths, and execution times is presented.

With the book, two appendices are provided. The first describes each of the pins on the various CPUs discussed in the book, while the second describes some of the support ICs available for the 65XX line of CPUs. If you are electronically inclined, these chapters will prove useful when interfacing to any of the CPUs described.

Although not part of the book itself, Western Design Center includes a current set of data specifications on one or more of its CPU line in the book's 3 ring binder. Although the information is somewhat redundant given the content of the book, readers can use the data sheets as an additional reference to double check accuracy of the book. It can also be used as a quick programming reference, as the 571 page book can prove somewhat unwieldy to use when developing applications.

The usefulness of this book depends on your needs. If you are learning to develop 65XX applications for the Commodore computer, the size and wealth of information in this reference is overkill. In addition, the extra information on the 65816 and 65802 will simply confuse you. However, if you are gearing up to develop 65816 native mode applications for the SuperCPU or other '816 equipped accelerator cartridge, you can find no better reference. Although the data sheets on the '816 provide some information on programming the CPU, the information is sketchy and incomplete at best. The data sheets are meant to provide basic information, not substitute for a programmer's guide. So, before you start developing those native mode apps for the CMD SuperCPU, grab a copy of this book to understand the features available to you.

-Jim Brain

CW & RUN BACK ISSUES

For years, *RUN* Magazine provided Commodore Users with a great source of information, and now CMD has given you *Commodore World*. Don't let this valuable information slip away—fill in the voids in your library now!

RUN Magazine Back Issues

Any 3 issues for $12.00, any 6 for $18.00, or any 12 for only $24.00

RMJAN89	January 1989	RMJUN90	June/July 1990
RMFEB89	February 1989	RMDEC90	December 1990
RMMAY89	May 1989	RMJAN91	Jan/Feb 1991
RMJUN89	June 1989	RMJUN91	JuneJuly 1991
RMJUL89	July 1989	RMJAN92	Jan/Feb 1992
RMAUG89	August 1989	RMMAR92	March/April 1992
RMSEP89	September 1989	RMMAY92	May/June 1992
RMOCT89	October 1989	RMJUL92	July/August 1992
RMSP89	Special Issue 1989	RMSEP92	Sep/October 1992
RMJAN90	January 1990	RMNOV92	Nov/December 1992
RMMAY90	May 1990		

Shipping: 3-6 issues - U.S. $3.00, Canada $5.00, Foreign $15.00; 12 issues U.S. $5.00; Canada $7.00; Foreign $20.00.

Commodore World Back Issues

$4.95 each, or any 3 for only $12.00

CW1	Commodore World Issue 1	CW9	Commodore World Issue 9
CW2	Commodore World Issue 2	CW10	Commodore World Issue 10
CW3	Commodore World Issue 3	CW11	Commodore World Issue 11
CW4	Commodore World Issue 4	CW12	Commodore World Issue 12
CW5	Commodore World Issue 5	CW13	Commodore World Issue 13
CW6	Commodore World Issue 6	CW14	Commodore World Issue 14
CW7	Commodore World Issue 7	CW15	Commodore World Issue 15

Shipping: U.S. and Canada $2.00 for first issue, plus $1.00 per additional issues.; Foreign $5.00 per issue.

TO ORDER CALL 1-800-638-3263

INTERCORP COMMUNICATION Presents

CSOFT Wares!

{As Seen on Many BBS's}

Digi<-->Dox V1.0
a SEQ. file reader that will read text as well as art,and supports drive' 8-11.It also has many Digi-sounds which are very entertaining.
$7.00 + $2.00 S&H
D.S. II "Cartoons"
10 of your Favorite Carteen Digi Samples on one disk ,with a very eas to use Menu System for loading and playing.
Great For the Kids! $3.00 + $2.00 S&H
Send orders to
Intercorp Communication 100 Pinegrove,Pontiac,MI, 48342
Alaska,Hawii,& International add $1.00 to S&h

CPU

P.O. BOX 1817
SHELTON, WA.
98584

The Commodore 128/64 Power User Newsletter

CPU is a quarterly publication that for two years now has kept its readers abreast of new developements and helped those to get more out of their investments.

Sample/Back Issues: $3.50
6 Issue Subscription: $15.00 us
International: $27.00 us
Canada and Mexico: $21.00 us

Ushering the Commodore user into the 21st Century.

The SysOp's Corner

By Max Cottrell
(mcphoto@izzy.net)

CREATING ANSI SCREENS

Notice of Disclaimer:
Many of the ideas and instructions used in this column are specific to a single Bulletin Board Program. Many things may work with other programs, and many may not. Some of the topics that I touch on are a direct result of over 3500 hours working with several computer configurations, and up to eight modems. Some of my ideas may not work with your configuration. Before you try any of them, be sure to check with your manual, or whomever is providing you with customer support. Those who assisted in the background information for this column:

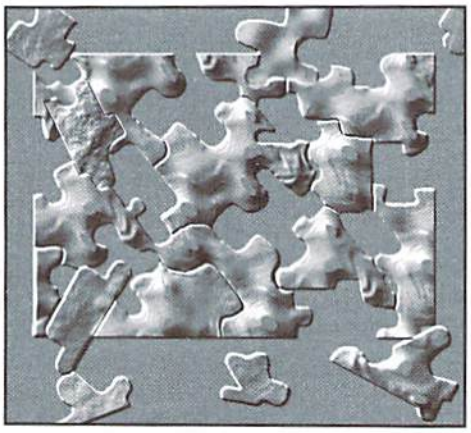
Andrew L. Fish, Marquette University, Michigan (list of ANSI commands)

Andrew Bernhardt, author of Does The Job BBS

Ali R. Talaie, author of Computer Acronyms, first edition

Sanford Walke IV, Isthmus corp. SYSOP (provides UNIX command help)

This particular column will continue with basic instructions for use of some of the ANSI codes that you can use on your BBS to create graphics, menus and screens that will keep your users interested in what you have to offer. I know that a lot of BBSs for the Commodores use the native graphics that Commodores use, so I will just give you some of the basic ANSI commands right now, and if there are enough requests for more,



then I will get into them in more detail in a future column. Before I continue, I think it's important to touch on another item.

Most of my company's presence is on the Internet: my photography company and my newspaper publish on a daily basis there. As a result, I am a regular maven in the newsgroups. My favorite of which (of course) is comp.sys.cbm. Lately, I've noticed that BBS programs are a hotly debated topic. If you want to start a debate, mention a BBS program, then stand back. Each programmer has worked hard to create a great program, and often each one has its own features (and quirks) that sets it apart from the others. Each SYSOP, potential or running a board, has a horror story to tell.

There seems to be a certain amount of difficulty in finding just the right BBS program, and when you decide which one you want, actually getting the thing to disk drive can be the hardest part of getting a BBS online. After seeing a lot of ground teeth lying around the newsgroups, I have set out to find the various programmers and software distributors that deal with BBS programs. Beginning with my next column, I'll try to highlight a few.

The Fun Stuff

O.K., onto the fun stuff. We'll create a basic text file that will allow you to see some of the cool stuff that you can do with graphics and commands. One of the things that you should consider is at what modem speed the user will be logging on. Sometimes if you are doing an animation, a 2400 baud user will not see as good of an effect as a 14.4 baud user. Likewise a 14.4 user might blink and miss a 20 frame animation.

ANSI graphics are a powerful tool that you can use to create some pretty neat things on the users screen. If you have the time and patience, you can really get some wild effects. Most of what I have done on my BBS is information based (as in no games or fun stuff) so I really have to keep hopping to keep a user's interest. The part that I like the best is the ASCII artwork that artists around the world create for BBS SYSOPs to use. I mentioned in my last column that I am setting up my BBS to allow you, the reader, the ability to see what I am talking about online while reading about how to do them here. I also have a nice gallery of artwork. I have used most of them on my BBS at one time or another, and like them, so I am passing them along to you. Unfortunately, these things can be complicated, so I won't type any of them in here. Doug is probably pretty happy about that!

Instead, if you email me, I will give you the logon instructions. I have set up the system to demonstrate how to make a menu look good to the user, and how to give the user what they want without asking, along with giving you some ASCII art that will work on just about any BBS.

ANSI was designed to be used on all computers, but has been associated with IBM computers. IBM (or a clone) manuals come with a listing of ANSI commands and how to use them. But, where ANSI really shines is in telecommunications. You can see some pretty cool graphics on various BBSs that use ANSI. This is because of the character control that it gives you when you are using a text based telecommunications program. You can think of ANSI graphics as a poor cousin to HTML (HyperText Markup Language). The United States Government has a BBS for federal jobs listings that uses a lot of these graphics. Even on my Commodore equipped with a terminal program like Novaterm and a low budget color monitor, you can see the most advanced of them. Some BBS programs will use PETASCII graphics and not ANSI. Here's another bit of trivia for you. If you have a shell account on an Internet Service Provider, the menus that they use also use ANSI commands. The whole idea behind ANSI graphics is cursor and color control. I also use ASCII art, which is keyboard characters arrainged into a picture. Some of these can be really detailed. However, they almost always require a black background and screen clears to view them properly, which is where we should start.

The Structure of ANSI

ANSI, like any other language for computers (even though its uses are for textfiles and animation, you can consider it a language of a sort), has specific families of commands. Let's look at color control. Here are some of the ANSI commands that are available, thanks to Mr. Fish.

How to Input the Commands

As you read my instructions on using graphics, you'll see (esc) used a lot. This means that you must use whatever code that your BBS uses for an escape sequence. This tells the computer that you are giving it a command. The BBS program that I operate uses CTRL P (holding down the Control key and pressing the 'P' key at the same time), or Commodore P to tell the system that an ANSI or some other command is about to be issued. The CTRL P is part of the escape coding from MS-DOS (Microsoft DOS—if you look at your C128's 40- or 80-column start-up screen, you will see the Microsoft copyright). As we all know, Microsoft provided Commodore with some of the DOS that we use on our machines. IBMs will have you press the escape key to send an ANSI command. Commodore 64s don't normally have an escape key, so only part of the command, CTRL P, or Commodore P is used. This is not true in all Commodore programs. As another example, if you were using a UNIX system (some IBM owners and most Internet providers use UNIX or a newer version called LIUNIX), the escape sequence would be CTRL V.

Cursor Control

Controlling your cursor is a very importing activity. With ANSI, you can make it go up, down, sideways, just as if you were using the cursor keys in a print statement on a Commodore. Well, you are doing this on a Commodore, you are just using a different route to get the same results. Here is a list of the cursor commands that should be available to you. Note that 'x' is a number.

COMMAND	DESCRIPTION
[xA	moves cursor up x lines,same column
[xB	moves cursor down x lines, same column
[xC	moves right x columns, same line
[xD	moves left x columns, same line
[x;yH	moves cursor to location x,y on screen
[xX	erases spot that cursor is on and x characters

This is *not* a complete listing of the text control commands. These are the ones that I have used with my system, so I know that they work. You can get some neat effects with these commands. One of the things that I do on my system when it is time for the user to input a system command (log off, download, etc.) is draw a reverse line of spaces, and then move the cursor back to the beginning of the line. When the user types in a command, the type shows up as non-reversed letters. It's a little thing, but looks great on the screen.

Colors

The background colors are separated from foreground (text) colors by the numbered prefix in the command. The use of an 'm' (note that it's a lowercase m) will tell the BBS that you are issuing a color command. The foreground is indicated by a '3' and the background a '4.' Here are the color commands:

FOREGROUND	COLOR	BACKROUND
30	black	40
31	red	41
32	green	42
33	yellow	43
34	blue	44
35	magenta	45
36	cyan	46
37	white	47

You can see that the last digit of the color commands are the same for both the foreground and background. Only the 3 and 4 are the different. Keep in mind that an 'm' is used at the end of the command. An example of this command might be: (esc)[40m. This turns the background black.

Attributes

Attributes are things like reverse text, blinking and underlining.

COMMAND	ATTRIBUTE
[0m	All attributes off
[5m	Blinking on
[7m	Reverse on
[25m	Blinking off
[27m	Reverse off

Clearing the Screen

When creating a menu or textfile, you need to decide whether or not you want to clear the screen. If the file has a lot of text or graphics, then I usually issue a screen clear. If there are just a few lines, then I might just use a few carriage returns to separate the lines of text. An example would be if you had a game that you created. The top half of the screen can be left alone for a graphic or a menu, and using cursor commands you can just add prompts at the bottom of the screen, changing the picture as needed. The biggest things is whether or not the previous screen needs to be on the monitor still. For instance, clearing the screen after letting the user see a disk directory is not a good use of this command. Clearing it for the Command input menu that your BBS uses is. Sometimes. The final decision is up to you, the SYSOP. This is another time where the beta tester and logging on from another computer come in handy.

You'll notice that ANSI commands are usually an escape key input, followed by a number(s) and then a letter. The letter part of a ANSI command is case sensitive, which is most important because an upper case J will do something different than a lower case j.

To add a screen clear to a file, we must use the command (esc)[2J. You can add a screen clear part way through the file. This can be of help if you are doing some kind of animation, or want to have multiple on screen prompts within a single textfile. The addition of the [character is also from our old friend MS-DOS. You should note that in an animation sequence, each code should have the escape sequence, even if the system allows for multiple ANSI commands. Mostly, it is for neatness of the coding, but having a single escape key sequence can sometimes have poor results. Don't ask me why, I just heard that from a UNIX guru.

So, to start our textfile, we will enter our first ANSI command!

```
(esc)[2J
```

This will clear the screen. Now, let's assume that you will be showing the user a textfile that will show them the system specifications of your BBS.

The next thing that we should do is to use a color command to make the screen background black so the the art will show up clearly.

```
(esc)[40m
```

Now we have a clear black screen. But what about the text color? How about a nice red?

```
(esc)[31m
```

There. Now we can add some specs about our BBS. Let's assume for the sake of this article that we are doing this on my BBS. You can change the text to suit your own system. I'm also going to add some other commands that we'll talk about next, so the full listing in the sidebar, "ANSI Info Screen".

As you can see, there are a lot more commands in this little bit of text than you might think. What we just did was to reverse the text (esc)[7m and change the text colors (esc)[3#m so that what we want highlighted will be in reverse. You also have to make sure that you change the screen back to the black background, red text when you are done by using (esc)[0m (turn off all attributes) and (esc)[31m after EACH series of text. To answer the question that all newcomers have to this, yes it is a lot of work. That is why you should only do the fancy stuff with files that will stay the same most of the time such as command menus, help files, etc... As far as shutting off the attributes (colors, reverse text) every time I change some aspect of the file, I do it from personal habit, not because you have to do it that way. Some will tell you to do this a little differently, but I like it because it makes sure that you have the text and background set correctly.

You will probably gravitate toward some ANSI commands that appeal to you more than others, which is where the personality of your BBS will come to life. After all, it is how a system looks to the user, whether it is an Internet Provider running on a SUN and UNIX or a Commodore 64 running a BBS and using ANSI, that shows the world the SYSOPs personality.

So, we have cleared the screen, changed the text color, and made some parts of sentences a reversed screen. You should be able to figure out how the rest of the commands work without too much trouble. A complete list of the ANSI commands is one of the things that I am planning on putting on my BBS. All of the commands follow the same rules and techniques for inputting them.

I'll leave you with a description of the technique that I use to create my files and animations. First, I log onto my BBS from my other computer. Then I use the editor in the BBS to create the file while online. After each line, I test the file to see if it is working correctly. One important thing to remember about ANSI is that the commands are not seen when the file is listed. So, writing down what you are doing on a sheet of paper is pretty important. Why? Because the command is not visable on the screen once you list the file. In fact it is very important. I know, because I can forget what I coded and when I go back to change something, I forget things like the number of columns I moved the cursor. This particular edition of the SysOps Corner should have given you a pretty good idea on using these commands. ANSI can be a powerful tool if it is used right. But in the wrong hands...

Next time, I will start with my big 'Interview the Programmers' project which should be entertaining to say the least. Until then, you can email me with questions, flames, input, ideas or even log on instructions.

ANSI Info Screen

```
(esc)[2J
(esc)[40m
(esc)[31m

Welcome to the (esc)[7m (esc)[36m
SySops Corner Demonstration BBS (esc)[0m (esc)[31m

System specifications:
Computer: (esc)[7m (esc)[31m
Commodore C128D (esc)[0m (esc)[31m
Drives: (esc)[7m (esc)[36m
FD2000, HD-20, RamLink (4 meg) (esc)[0m (esc)31m
Modem (esc)[7m (esc)[32m
US Robotics Courier 14.4 with ASL/V.32 (esc)[0m (esc)[31m
Todays date and time: (esc)[7m
( Today's time and date )(esc)[0m (esc)[31m
```

A Grower's Guide to User Groups

The President of the Tampa Commodore User Group shares his group's methods for increasing membership

As an old 'die hard' supporter of Commodore and active user group president on and off for the last ten years, I've seen Commodore go up and down and up again as we move through time. As with all things, times change, as so the struggle to survive presents itself as a constant need to adapt. Commodore is still a viable machine as it was way back in 1982. Only the nature of the computer has changed. The users of Commodore are old 'die hards' and still a new large array of first time computer owners. I've been in contact with several other sister clubs, all experiencing the problem of loss of members. So it may be true, except for us in Tampa. Keeping abreast of changing times, we have realized that our mission as a support club has also changed. In fact, we are more important than ever. Commodore is still a hot machine, only the price has changed from in store purchase new to used. In many cases the first time computer owner, happy at finally being a computer owner, suddenly realizes the support has disappeared off the open market. This, of course, is not true. The support has just moved from local stores to mail order. New programming and hardware advances has made the C64 and C128 still the best home computer, in my opinion, of anything around.

Here in Tampa, we see our mission as one to pickup the 'stragglers'. Oh yes, Commodore users are still out there, but do they know that your local Commodore support group is there? As a club, we have all the support companies on file to service new members. And we have many new members. In fact, we have tripled our membership since the institution of new policies since January 1996. And we shall quadruple or more in size before this year ends, reaching our former strength not seen since the late 80's.

Here is the six part plan we instituted:

(1) STORE FLYERS - If you have a local store available, print up some detailed in-store flyers advertising your club. This will become your main source of growth.

(2) BUSINESS CARDS - Have business cards printed up to advertise your user group. On ours, we have our club name and a few lines such as "C64 and C128 support", "programming", "large software libraries", and "printer re-inking". We also have "more information" and the phone number of the new member coordinator. These business cards can be put up on store bulliten boards, handed out to flea market dealers, at garage sales selling Commodore equipment, taped onto equipment at thrift stores and alike, and passed unto sellers of equipment advertised in local papers. These business cards are welcomed as it helps sell the Commodores being offered for sale. The first thing that the new owner will do is to contact your club for support and information.

(3) LOCAL BBS - If you have a local Commodore BBS in you area, align your club with it. Many times most on line users, for whatever reasons, don't participate with their local club. But here in Tampa, we have merged the two concepts together. As a club, we even have private message bases and downloading of club owned public domain software libraries. We have also designed the club to conduct on-line business for those members unable to make a voting meeting. We pride ourselves with BBS get togethers quarterly where we litterly take over a local resturuant establishment for a Sunday noon brunch with lively conversation on the latest Commodore upgrades and software.

(4) LOCAL COMPUTER PUBLICATIONS - Local computer magazines and such are probably offerer free and distributed widely among local computer stores. Check these out for free advertisements of your user group. You may be surprised of old Commodore owners, who have lost touch and out shopping for a new system. You may be further surprised that they are more interested in upgrading their Commodore system rather than to invest more capital into a new one. Remember, you club has the listing of Commodore support no longer seen in computer stores.

(5) LOCAL NEWSPAPERS - Check out your local neighborhood newspapers for free advertising of you user group. You can get free news print under the Weekly News or Club Information sections. You may have to send weekly letters or fax the club meeting times or place to keep this kind of advertisement active.

(6) CABLE TV ACCESS - All cables companies offer a free local access channel. Your cable company may also offer a channel free of charge advertising local clubs and community events. Design a club ad and have it on the tube. We have had several contacts and new members through this kind of advertisement.

Oh yes, Commodore has changed! But so has the world and all computering. All the above ideas have worked successfully for us. Try it in your local support group. Only a few local users and no organized support, form a local group, advertise, and watch it grow. Commodore is still around. In most areas, users are unaware as to how much it has been upgraded in hardware and software. Computering is still a fun hobby in the home, with great educational and in-home business applications. The mission and life blood of Commodore users groups as we reach into the millenium is to reach out to the silent army of 'stragglers' and new first time owners, excited about their first affordable home computer. My words to you are "ADVERTISE AND GROW". Can you beat our club growth rate as we have experienced here in Tampa?

Jayme Rice
President, Tampa Commodore Users Group

Carrier Detect

By Gaelyne R. Gasson

TRANSFERRING FILES ON THE INTERNET WITH FTP

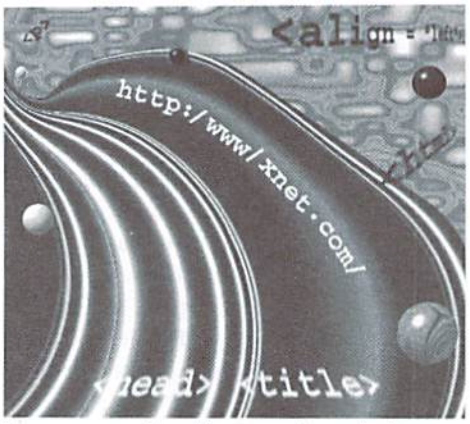
As Easy As FTP

A recent advert for the Internet grabbed my attention with big bold lettering: "Welcome to the smorgasbord!". It went on to describe things they had to offer, including the ability to download files from hundreds of Internet sites via FTP. What's FTP all about? It sounds mysterious. It uses Unix commands. It must be terribly difficult to do, right? Wrong.

What is it?

FTPing is similar to being able to view BBS download directories without the benefit of having a program to give you the file descriptions. The basic gist of it goes something like this. First you log onto an FTP site, (usually anonymously). Then, just as you would with your Commodore computer when looking for a particular program, you move about the directories, listing them. When you find what you're looking for (or something that interests you), you send the file to your home directory on your Internet Provider. Once the files are in your home directory, you can download it to your computer. Some FTP systems let you skip this step and download files to your computer without exiting the FTP program. That's not too difficult is it? What's that? You're worried about all the little steps between and what commands you need to use? That's understandable. I felt the same way the first time I read about FTPing. Let's walk through the process and discuss it along the way.

FTP is an acronym for "File Transfer Protocol", and it's a utility, not a command. Once you start it, it's in control until you exit or quit the utility. How you go about starting a session depends on the type of Internet access you have. If you're using an Online service, or your provider has a menu, you will find FTP as a menu option, otherwise you will be using a Unix shell. Although



it may seem easier with a menu, it may be faster to use the shell instead. Another advantage of using the Unix shell is that most text and commands will match articles, such as this one, or books you read on the subject. When you FTP using a menu, you're at the mercy of whomever wrote the menu program for the type of prompts you get. I know it's a weird type of choice to make—no prompts at all versus prompts that might not make any sense. Welcome to the Internet. <smile>

Anonymous Logins

We're going to jump in via the plain FTP utility from a Unix shell. There are two ways to start FTP. One way is to type "ftp" on the command line, but this isn't very efficient since you will next be telling the utility where you want to go. One Internet site with heaps of Commodore files is ccnga.uwaterloo.ca, so let's go check it out. On the command line, type:

```
ftp ccnga.uwaterloo.ca <return>
```

My first attempts at FTPing were frustrating for me because the system kept telling me no such place existed. After several tries, I finally found I was typing with dyslexic fingers, and I'd transposed two characters. (I'd been typing "gna" instead of "nga"). Once I typed the name correctly, everything worked fine. If you misspell the initial address, you will find that you're already within the utility, and can tell this because your prompt will have changed to "ftp>". From this prompt, you must tell it to "open" the site you wish to go to. For example:

```
open ccnga.uwaterloo.ca <return>
```

Once you've connected, you should see something like:

```
Connected to ccnga.uwaterloo.ca.
220 punisher FTP server (SunOS 4.1) ready.
Name (hal9000.net.au:moranec):
```

Your Email address will appear in the parenthesis, and a cursor will be apparent right after the colon. If you type in your name here, the system assumes you're a pre-registered user of ccnga.uwaterloo.ca and will expect a real password. Instead, type the word "anonymous" and press return. You'll be prompted for the password:

```
331 Guest login ok, send ident as password.
Password:
```

Translated to English, "send ident as password" means "type your Email address for your password". As you type, you'll notice that you can't see what you're typing, as it doesn't "echo" your password. So type carefully. If you define a macro in your term program with your Email address you can enter this very quickly without errors.

If you mistype "anonymous" or your Email address, you may be able to continue with logging in by typing "user anonymous" and pressing return. This doesn't always work with every Internet site, but is worth trying. If you've made a typing mistake, it's possible that the remote system (in this case, ccnga.uwaterloo.ca) might "close the connection" and you find yourself looking at the FTP prompt on your own system. Don't panic. Try again, but type more carefully or define macros in your term program to do the typing for you.

On slow systems, it can be a long time between typing a command and something happening. Have patience. Don't keep pressing keys, that will only make it more confusing for you. Once you've successfully logged in, you'll see:

```
230 Guest login ok, access restrictions apply.
ftp>
```

Your cursor will be next to the "ftp>" prompt.

Moving Around and Seeing the Sites

From here you use Unix commands to navigate. You'll have to move to the area where the Commodore files are held. On ccnga.uwaterloo.ca you'll want to move to the /pub/cbm directory. The command to change directories is the same as you use on your Unix account: the cd (change directory) command. Type:

```
cd /pub/cbm
```

Unlike Unix, when you send a command that works the FTP utility returns a message such as:

```
250 CWD command successful.
or 200 PORT command successful.
```

After changing directories you'll want to look at what's in it so you can decide what goodies to download. This is where FTP takes a different turn from Unix, as you can type "dir" and press return OR use the Unix commands:

"ls" or "ls-al" (see Example 1).

This list is abbreviated to save space. In fact, it's so long, it will scroll off your screen! You can send a command that will show the directory and "page" it to the screen, so you can use the space bar to scroll each screen. The command is:

"ls -l lmore"

Add this to your term program's macros and you can hit the macro key instead of typing it each time you need it. A similar command is "ls -C", this will show you the files and directory names in multiple columns but you won't have the added information about each file. To stop a paged listing so you can send other commands while seeing a filename, press the CTRL and 'c' keys simultaneously.

Speaking of info, just what ARE we looking at? This is a Unix directory listing. Working from left to right, the left-most column shows the permissions" for each file and directory. If the very first column has a letter "d" in it, that entry is another directory. In the above example, there are two files, and the rest are directories. The next column relates to the something called the "link count", and we can ignore it. Most Commodore users who do much telecommunications will recognize the name shown in the third column. Craig Bruce is the administrator for this site, and he is the owner of the files and directories. In other FTP directories on ccnga.uwaterloo.ca, you will see "ftp" or "nobody" in this column. Just after this you'll see the size of the file or directory in bytes. The date displayed shows when the file or directory was last modified. This can be useful if you're planning to use FTP to stay up to date with the files. Many people check the INCOMING directory daily, and check the date shown next to the directory name to see when new files have been uploaded. The last item is the file or directory name. This is Unix, so the filenames are case sensitive. Type the filenames as they appear in the listing.

The first file in the directory is "00README". You can read this by typing: "more 00README". It will display one screen at a time, and you use the space bar to see the next. Any text file can be read this way, which is useful for reading file descriptions, or the rules for using an FTP site.

Let's check out the INCOMING directory and see what's new. To do this, we move to the directory with the cd command we used earlier. Type: "cd INCOMING". Use the "ls -l lmore" command to list the directory (see Example 2). Here you see the same names as were in the main /pub/cbm directory. To save himself time, Craig set this area up so those who upload files can put them in the appropriate INCOMING directory. This way, after he's checked the files, he can move them to the directories that fit the file category. In other words, this is a temporary holding place for files.

EXAMPLE 1

```
200 PORT command successful.
-rw-r-r-     1 csbruce           5331 Apr 25  1994     00README
drwxr-xr-x 21 csbruce             512 Nov  6 19:38     INCOMING
drwxr-xr-x  3 csbruce            1024 Nov  6 19:55     archivers
-------------------------------------------------------------------
-rw-r-r-     1 csbruce           2048 Nov  6 22:41     ls-lR
drwxr-xr-x  4 csbruce            1024 Jun 15 11:28     telecomm
drwxr-xr-x  4 csbruce            1024 Jul 26 20:42     util128
rwxr-xr-x   5 csbruce            1536 Nov  6 22:23     util64
drwxr-xr-x  2 csbruce            1024 Nov  6 22:25     vic-20
```

EXAMPLE 2

```
INCOMING:
total 25
-rw-r-r-     1 csbruce           2145 Oct 22 18:23     00README
drwxrwxrwx   2 csbruce            512 Nov  6 19:55     archivers
-------------------------------------------------------------------
drwxrwxrwx   2 csbruce            512 Dec 25 05:15     telecomm
drwxrwxrwx   2 csbruce            512 Jan  1 13:42     unix
drwxrwxrwx   2 csbruce            512 Feb  6 18:29     util128
drwxrwxrwx   2 csbruce            512 Feb  3 09:37     util64
drwxrwxrwx   2 csbruce            512 Nov  6 22:25     vic-20
```

EXAMPLE 3

```
cd telecomm
dir
INCOMING/telecomm:
-rw-r-r-    1 csbruce           47872 Sep 16 01:42     des200-1.sfx
-rw-r-r-    1 csbruce           28160 Sep 16 01:43     des200-2.sfx
-rw-r-r-    1 csbruce           43776 Sep 16 01:44     des200-3.sfx
-rw-r-r-    1 csbruce           20736 Sep 16 01:45     des200-4.sfx
-rw-r-r-    1 csbruce           32768 Dec 25 05:37     wavedemo.sfx
-rw-r-r-    1 csbruce             366 Dec 25 05:37     wavedemo.txt
```

Let's check out the INCOMING/telecomm directory. First we change to the directory with the cd command, then we list it (see Example 3).

Again, this directory has been shortened and changed for demonstration purposes. Someone uploaded the demo of Maurice Randall's "The Wave" terminal program on Christmas—and it has a readme type file. You can find out what it's all about by typing: "more wavedemo.txt".

Now that we've found the files we want, let's grab them, and I'll show the different commands used when getting a text and binary file.

Getting Files

FTP uses different protocols when sending text or binary files, so you must tell it which one to use whenever you get files. Send the command: "ascii". You'll see a message that says "200 Type set to A". (The A is for ASCII). Type: "get wavedemo.txt", and the system will respond that it's opening an ASCII transfer. You won't see much until it finishes the job, but since the text file is only 366 bytes, it won't take very long at all and you'll see something like:

```
local: wavedemo.txt remote: wavedemo.txt
200 PORT command successful.
150 Opening ASCII mode data connection for
wavedemo.txt (366 bytes).
```

When finished, it will show:

```
226 Transfer complete.
376 bytes received in 0.18 seconds (2.1 Kbytes/s)
```

Now we're ready to get the SFX file. Type: "binary" and press return. This time the message you'll see is "Type set to I". The 'I' stands for "image". I know, it seems like it should say "Type set to B" (as in Binary), but this is Unix where not everything appears to make sense. Type: "get wavedemo.sfx" and you'll get a message that it's opening a binary transfer, and after a few minutes you'll get a message back similar to the one above.

You can't hurt an ASCII file by transferring it as binary, but you WILL have problems if you transfer a binary file using the ASCII protocol. Always remember to set the type to binary before transferring a program or other binary file. How do you know the difference between binary and ASCII? Binary files are any that you would load and run, such as programs or any files that are archived. ASCII files are text files and usually end with the letters '.txt' or .'doc', or they have names that are dead give-aways like "README".

What if you want to get several files at once? Easy. We can use the mget command (the 'm' means "multi") and a wildcard. For instance, if you want the Desterm files shown above, you would start by typing "binary", then "mget des200*". Before transferring each file you'll be asked "mget des200-1.sfx?". Answer each prompt with the letter 'y' for yes or 'n' for no (if it displays a file you don't want to transfer).

Once you're experienced with using Unix wildcards you probably won't want to baby-sit the process and answer prompts for each file. To avoid this before using "mget", use another command called "prompt". This will turn off the questions and the files will be transferred without any stops between each file. Another useful command is the hash mark (#). The hash lets you see what's happening as the files are transferred. If you give this command prior to beginning a transfer, you'll have something to watch, and an indication the system is indeed doing something. Both prompt and the hash are toggle commands —you turn them on and off by using the same command. You need only use these once during a session. For example, if you're downloading the above files and wish to turn off the prompts and have it display information during the transfer, you would type:

```
prompt
#
binary
mget des200*
```

Sometimes getting files via FTP can be extremely fast and efficient. Other times it can be a slow process. This depends on the speed of your Internet provider and how many users are online, along with the FTP sites' speed, and how many other users are accessing it. Also, a lot depends on WHERE you are FTPing to. It's best, if possible, to use FTP sites that are nearer to you geographically, as there is a noticeable lag time when transferring files from one side of the world to the other.

We're done with using the FTP utility so we can exit it with the "quit" command. Wait a minute, I hear you saying, "We didn't download anything yet!" You're right, we didn't, but we will, once we're back to using our own system.

Back "Home"

List your home directory and you'll see that you have the wavedemo.sfx and text file, along with the Desterm files. To download the files all at the same time using Novaterm's Zmodem, the command is "sz wave* des200*". For Ymodem batch, use "sb wave* des200*". If you're using Xmodem, you'll have to download each file individually using the "xb" and typing in the individual filenames. You can cheat a bit by using just enough of the file name so the system knows which one to send, such as "wavedemo.s*". Once you've downloaded the files, if your system has

FTP COMMANDS

Start an FTP session	ftp <sitename> or ncftp <sitename>
Login as	anonymous
Send ident as password	your Email address
Change directories	cd <dirname> or <dirpath>
View directory	dir ls ls-al
w/paging	ls -l lmore
columns & paging	ls -C
Stop dir list	CTRL-c
Read textfile	more <filename>
Set transfer to ASCII	ascii
Set transfer to Binary	binary
Suppress prompt toggle	prompt
Show progress	# (hashmark or American pound symbol)
Get one file	get <filename>
Get multiple files	mget <filenam*> or mget <file1 file2>
Put file on FTP site	put <filename>
Put multiple files on site	mput <filename>
Quit FTP utility	quit, exit or control c.
Download using Zmodem:	sz <filename(s)>
Download using Ymodem:	sb <filename(s)>
Download using Xmodem:	xb <filename>
Delete files after download	rm <filename>

tight restrictions on how much hard drive space you're allowed to use, you'll have to delete the files from your home directory right away. If you don't have strict limitations, leave the files so if you have any problems you can try them again without the need to FTP them again.

Once in awhile you might download a file that doesn't work. It won't dissolve or run, and you've tried just about every possibility you can think to try. Recently, I was alerted that a file on my FTP site at hal9000.net.au had a problem. After some searching and comparing, I found I'd made a mistake and forgot to type the word "binary" before "putting" the file on the site. Because of this the system assumed it was text and anyone who downloaded the file couldn't dissolve it. If you've downloaded a bad file and know the name of the uploader (or at very least the name of the person in charge of the FTP site) send them Email and let them know what kind of problem you are having. If the file is bad it can be replaced or deleted.

COMMODORE RELATED FTP SITES

Host:	Directory:
ftp.rrz.Uni-Koeln.DE	/usenet/comp.archives/auto/comp.sys.cbm
src.doc.ic.ac.uk	/usenet/comp.archives/auto/comp.sys.cbm
rtfm.mit.edu	/pub/usenet-by-group/comp.sys.cbm/main-faq
cs.dal.ca	/pub/comp.archives/comp.sys.cbm
ccnga.uwaterloo.ca	/pub/cbm
ftp.cs.tu-berlin.de	/pub/c64
ftp.armory.com	/pub/user/spectre
hal9000.net.au	/pub/cbm
nexus.yorku.ca	/pub/Internet-info
ftp.demon.co.uk	/pub/cpm
ftp.inf.bme.hu	/pub/cbm/
ftp.cs.columbia.edu	/archives/mirror1/kermit
plaza.aarnet.edu.au	/pub/kermit/c
flubber.cs.umd.edu	/rec/newballistic
nic.switch.ch	/mirror/kermit/bin
ftp.gmd.de	/if-archive/games/c64
wilbur.stanford.edu	/pub/emulators/c64
syrinx.umd.edu	/rush/systems/c64
ucsd.edu	/midi/software/c64
ftp.hrz.uni-kassel.de	/pub/machines/vic-20
cs.dal.ca	/comp.archives/c64
wuarchive.wustl.edu	/mirrors/cpm
frodo.hiof.no	/pub/c64
bbs.cc.uniud.it	/pub/c64
ftp.rz.uni-hildesheim.de	/pub/c64/
ftp.funet.fi	/pub/amiga/audio/misc/sid-tunes
sol.cs.ruu.nl	/pub/MIDI/PROGRAMS/C64
	/pub/MIDI/DOC
oak.oakland.edu	/pub/cpm
	/pub2/cpm
watsun.cc.columbia.edu	/kermit2/old/c64
	/kermit/bin
x2ftp.oulu.fi	/pub/cbm
	/pub/cross
tolsun.oulu.fi	/pub/c64
	/incoming/c64
	/pub/c64
131.188.190.131	/pub/c64
	/pub c64/POLDI
	/pub/poldi
ftp.funet.fi	/pub/cbm
	/pub/misc/c64
	/pub/kermit/c64
ftp.eskimo.com	/u/v/voyager/Novaterm
	/voyager/Novaterm/Deutsche

Reversing the Process—Putting Files on an FTP Site

So how DO you put files on an FTP site? I gave the clue in the last paragraph. It's the reverse of getting files from FTP sites. First, upload the file to your home directory. Start FTP and go to the site and directory where you want to upload the file. Type "binary" or "ascii" depending on the type of file, and then use the "put" command to transfer the file to the FTP site. If there's just one file, type "put <filename>", or if there are multiple files the command is "mput file1* file2*". Before starting your FTP session, type a short description of the file or files and save it to your home directory, then put the text file with the description on the FTP site too.

NcFTP

Now that you know how to do FTP the old fashioned "hard" way, I'll let you in on a nicer utility. It's called NcFTP (pronounced `Nick-F-T-P'), you can use the same commands with it as FTP, but this utility automatically sends the "anonymous" text and your Email address as the password for you. It keeps track of what sites you've FTP'd to and when you use it again to return to a site it takes you back to the same directory you last accessed. When transferring files it displays bar-graph that's continually updated until the transfer is finished. It also gives an "ETA", or "Estimated Time of Arrival" percentage. You start NcFTP by typing "ncftp <sitename>". If your Internet provider has this utility available it will start, otherwise you'll receive a message telling you that it could not find the utility. After you've used NcFTP once, you can start it by typing "ncftp" and then type "open" once the utility has begun. It will show you a numbered list of FTP sites you've visited, and you can type the number item of your choice and it does the rest.

Wrap Up

Now that you know what FTP is all about, you'll be able to use it to get all kinds of goodies. Commodore programs aren't the only things you'll find on FTP sites that you can use. You can download FAQ (Frequently Asked Questions) and other text files that cover your interests, and you can download graphic images to view. Remember too — it's nice to share. If you have PD or shareware, artwork or text files of interest to others, share them by placing them on an FTP site.

GEO PROGRAMMIST

by Maurice Randall

Outside of GEOS, it is a simple matter to create a new file from within a program. You begin by sending a command to your disk drive to open a new file, and then send the data to the drive to be placed within that file. Once finished, you close the file and the job is complete.

But in GEOS it is a different matter, at least if it's GEOS type files that you are wishing to create. Now, I'm not talking about your actual applications that you create using geoProgrammer, I'm talking about having a need to create something like a data file while your application is running. As an example, when using geoWrite, it creates the data file that you are typing your text into. Sooner or later, you will design an application that needs to have the ability to create it's own data files like geoWrite does, or geoPaint, geoPublish, etc.

We already know that GEOS files are somewhat different from a normal Commodore disk file in that they have a header block attached to them and can also be in one of two forms, sequential or VLIR. The directory entry also contains additional information that you normally do not see in a Commodore disk file.

In the GEOS kernal, there is a routine called 'SaveFile'. This is the master routine that helps us to create files of whatever type we desire. If you have a need to create data files from within your application, this routine will help you do so. In the process, though, you will discover some quirks with geoAssembler concerning the header block. Just follow along and I will show you how to do it in a manner that works.

The Header Block

In geoAssembler, there is a directive that is used to create the header blocks for your GEOS files. It is described beginning on page 5-50. At times, it will seem like this directive is being used to create a mental block, rather than a header block. geoAssembler is very strict with this one. You must be very careful how you format the source code in between the .header and .endh directives.

You may have already used these to create the header file that is used when you assemble and link an application, but did you know that you can also embed these directives inside the source code that makes up your application? After all, when your application creates it's own data files, it will need to build a header block for each of those files. That means you will need a header block of some sort contained within your application. The header block is one of the main requirements of the kernal routine 'SaveFile'. Most of the info it needs is within this block of data. Don't get confused with these extra header blocks, they are not at all connected to the one that is used to create your application. They are an actual part of the source code that makes up your application.

Looking at Savefile

Let's take a look at the routine SaveFile before going any further. There are only about 3 things required here. First, as already discussed, you need a 256 byte area of memory that describes the header block. Then you must point r9 at this block of memory. In r10L, you load the desired directory page that you would like your data file saved to. Normally this should be a zero so that if there are any empty spots on the first page, that is where your file will go. If not, then the first available slot will get used. The GEOS Programmer's Reference Guide states this one incorrectly. It says that a 4 here will attempt to place the file on page 4 of the DeskTop, but it will actually go to page 5. Now, you simply do a jsr to SaveFile and your data file is created.

Now for the Fun Part

Let's dig into that 256 byte header block again. This is the most important thing. It's got to be right or your data file won't work. In fact, you might not even get geoAssembler to assemble it correctly inside your application to begin with!

Always remember, that whatever goes in between the .header and .endh directives will end up being exactly 256 bytes long. If the code you place there is less than this, geoAssembler will make up the difference with additional zero bytes. That way, you don't need to count what you place there. But what you 'do' place there better be in the correct order, or geoAssembler will make you do it over again until you get it right.

You can copy the example I have in the sidebar here and then make only the changes you need for your own purpose. Be careful not to change the order or the length of each item.

Let's look at each item in detail. I've begun by placing a label, 'StartOfHeader', at the start of our header block. You can change this to your liking. This can be used to make reference to any part of the header block once our application is running. If we want to change the comment that will appear in the DeskTop's info box, we can access that as StartOfHeader+160.

I should point out one of geoAssembler's limitations here. There is an equate for that offset to the comment area called 'OFF_GHINFO_TXT'. But I've run into problems at times when equates are used along with labels. Sometimes it seems to work and other times it doesn't. If StartOfHeader+OFF_GHINFO_TXT is referenced from within the same source code file, it might work. But if from a different file, then geoLinker must resolve the address and it may or may not do it right. You might get an unresolved error message when this happens. The cure at this point is to use the actual value, such as 160, instead of the equate. This was one of the bugs that was supposed to be fixed in geoProgrammer 2.0, but was never released.

Now we come to the .header directive. This let's geoAssembler know how to assemble the code up to the .endh directive. You will end up with 256 bytes between these two. The first two bytes of the header block are normally $00,$FF. But for SaveFile, we need to have a pointer here to the filename that our data file will be given. Here's another assembler bug. No matter what you put here, it will always end up with $00,$FF once it is assembled. But go ahead and put a pointer to a filename here anyway. It can be a reminder for our own purpose. I'll get back to this shortly.

The next two bytes describe the size of the icon. This never changes and is always 3 and 21. A photo scrap is then placed here. And no matter what you put here, only the first 63 bytes of image data will be used for the icon. But to be safe, always cut out a scrap from geoPaint that is 24 pixels wide and 24 pixels high. The assembler will ignore the lower 3 rows of pixels as it generates your header block.

Next we have 3 bytes for the Commodore filetype, the GEOS filetype and the structure type. For our example, we are creating a USR file, it is a GEOS application data file, and it is of a GEOS sequential structure. (We will get into VLIR data files next issue.)

Next comes 3 memory pointers. The first one points to the start of the data that we wish to have placed in the file. Next is a pointer to the last byte of the data. Our example is actually going to save an entire region of memory that is 1000 bytes long. Your own application can alter these pointers as needed. The third pointer is not important for a data file, but would be if our application was creating another application file. This would be the location that would be called when the file is first loaded. A data file is not run, so whatever is placed here is unimportant.

Next comes the permanent name for the data file. This one is up to you however you want to name it. Each data file you create will contain the same permanent name. This is how your application can have only it's own data files displayed in a file requestor box. geoWrite does this by naming it's files 'Write Image'. This has nothing to do with the filename that is in the directory. GEOS can examine this permanent name and single out only those files from the directory.

The permanent name must always be exactly 12 characters long. If it is shorter, then add additional spaces to bring it to 12 characters. This should be followed by a version number, as in the example and then 4 zeroes. For a data file, the 4th zero is not necessarily important, but it would be for other types of files. That byte identifies the type of computer system the file is intended for.

The next 20 bytes is for the author's name. Again, it is not needed for a data file to have an author's name. Normally, an application would have one, but this data file will most likely be partially created by the person that is using your application. So, it is purely up to you what to put here. But I have always found it to be a good idea to always put 19 printable characters plus one zero byte here. Fill those 19 bytes with spaces or with your own name padded with spaces if you'd like, just don't put all zeroes there.

Next comes the name of the parent application. In our example, you see the name 'OurApp   V1.0'. This would actually be the permanent name that would be listed in our own application's header block. If the user double-clicks on a data file, the DeskTop looks at this name and finds the application that contains this in it's permanent name string. It then knows which application to load. If you do not want to make your data files capable of being loaded in this manner, then fill these 16 bytes with spaces. Follow this also with 4 zero bytes.

Next, a simple .block 23 will make the assembler generate 23 zero bytes. This brings our header block up to a total of 160 bytes. Normally, the comment we want to appear in the info box would come next. But anytime I've used the .header directive in this manner, the assembler always gives me an error if I place a comment here. So, instead, we have to end our header at this point with the .endh directive. The assembler will put zero bytes in the block until it has filled it up to the 256th byte.

Once our application is running, we will have to manually move the desired comment into the header block in memory before using SaveFile. Also, since the assembler put a $00,$FF in the first two bytes, we have to redo that also. In fact, this should be done each time SaveFile is called. Otherwise, SaveFile will look at the address at $FF00 for a filename.

Here's the routine that will do all this for us:

```
LoadW r0,#dataInfo
LoadW r1,#(StartOfHeader+160)
ldx #r0
ldy #r1
jsr CopyString
LoadW StartOfHeader,#dataName
LoadW r9,#StartOfHeader
LoadB r10L,#0
jsr SaveFile
```

You will then have a data file created on the currently open disk. One of the things you will want to keep in mind is the part in the header block that identifies the memory locations of the data to be saved. If this area gets relocated or changes in size, you will have to change these pointers accordingly before calling SaveFile. Also, you will notice that I added some additional zeros to the filename at dataName. This way, we have a full 17 byte location allocated for a 16 character filename plus a null terminator. Your application might have a need for creating more than one data file and so the filename here would be changed accordingly.

If you follow these rules I have outlined, you should not have any problems. You can deviate slightly with some things, but watch out if you do. The assembler just might give you errors. Next time, we will find out how to alter the data that has been saved in a data file and also talk about creating a VLIR data file and working with it.

```
StartOfHeader:
  .header
  .word dataName
  .byte 3
  .byte 21

  [Place a photo scrap for an icon here]

  .byte $80|USR             ;CBM file type.
  .byte APPL_DATA           ;GEOS file type.
  .byte SEQUENTIAL          ;GEOS file structure type.
  .word dataArea            ;start of data.
  .word dataArea+1000       ;end of data.
  .word dataArea            ;init address.
  .byte "PermName    v1.0",0,0,0,$00
  .byte "Maurice Randall    ",0
  .byte "OurApp    V1.0",0,0,0,0
  .block 23
  .endh

dataName:
  .byte "SampleData",0,0,0,0,0,0,0

dataInfo:
  .byte "This is a sample from "
  .byte "Commodore World magazine.",0

dataArea:
  .block 1000
```

ASSEMBLY LINE

PROCESSOR STATUS REGISTER FLAGS

Jim Butterfield

Previously, we have used test-and-branch code in an intuitive manner. When we said, "CPX #$06 .. BEQ $2055", we meant "Compare the X register with a value of 6; Branch if Equal to address hexadecimal $2055". It was sensible and natural.

But the two instructions are separate and distinct. What links the result of the CPX (Compare X) to the BEQ (Branch Equal) instruction? Answer: the condition is recorded in the *Processor Status Register*. The CPX instruction marks its results into this register, and the BEQ tests part of the status register to see if a branch is called for. In particular, the BEQ instruction tests a bit in the register called the *Z flag*; but perhaps we're getting a little ahead of ourselves.

Having an instruction leave "condition bits", to be tested later, has interesting side effects. For example, the two instructions don't need to be consecutive. They could be separated, providing that the intervening instructions don't mess with the relevant status flags. Here's another possibility: if needed, we can carefully save the results of an operation—that is, save the Processor Status Register contents—and bring back the results later for testing. We'll explore both of these circumstances later.

The Four Flags

As Figure 1 shows, the Status Register has four flags which can be tested by means of "Branch" instructions. There are eight Branch instructions, and each of them tests one of the four status flags to see if it is on (set) or off (clear). Two interesting things to note about the 6502/6510 branch instructions: there's no unconditional branch; and a branch can jump only a short distance, somewhat over a hundred bytes forward or backward.

Some instructions affect no flags at all; some affect only one or two; and a few instructions (mostly arithmetic) can influence them all. We'll deal with the busiest flags first.

Z - The Z (or Zero) flag is a busy flag whose condition is affected by every instruction that modifies the contents of a data register. Thus, a Load instruction (LDA, LDX, LDY) will set the condition of the Z flag depending on whether or not the value loaded is zero or not. Note that the Store instructions (STA, STX, STY) do not modify registers, and thus don't affect the Z flag, or any flag, for that matter. After a comparison (CMP, CPX, CPY), the Z flag signals whether the compared values were equal or not. If the Z flag is clear, BNE (Branch not Equal) will branch; if the Z flag is set, BEQ (Branch Equal) will branch.

N - The N (or Negative) flag is also affected by any instruction that changes a data register, so it's another busy one. The term *negative* needs more explanation than I can give here: I'll just say that the N flag tracks the highest bit of the result; if the high bit of the data is set, the N flag is set, and vice versa. After a comparison, the N flag takes on a fairly complex status; don't use it for greater-than tests (use the C flag instead). If the N flag is clear, BPL (Branch Plus) will branch; if the N flag is set, BMI (Branch Minus) will branch.

C - the C (or Carry) flag is not affected by simple data operations such as loading a register, so it's less busy that the previous two condition flags. Part of the C flag's usage is with arithmetic. After an addition (the instruction is ADC, Add with Carry), the C flag is indeed a true carry indication: it means there wasn't room to hold the sum within a single byte.If you have a multiple-byte number, you'll just move along and add the carry into the next column (as we do with decimal addition). If all you've got is a single byte, there's no room for the result and you have an "overflow". The C flag is also involved in subtraction, where its role is that of an "inverted borrow" (don't worry about that one too much until we get to math stuff). With Shift and Rotate instructions, the C flag holds the bits that pop off the end of the data byte. Perhaps most importantly: after a comparison operation, the C flag is set if the register is greater than or equal to the value it's being compared to.

The C flag may be directly set or cleared by using instructions SEC (Set Carry) and CLC (Clear Carry). If the C flag is clear, BCC (Branch Carry Clear) will branch; if the C flag is set, BCS (Branch Carry Set) will branch.

V - the V (or Overflow) flag is affected only by addition and subtraction, and by the curious BIT instruction (whose main purpose is to test I/O chip status).Even with ADC and SBC, the V flag is usually only meaningful when the computer is dealing with signed numbers. A sidenote: although this won't happen in your computer, the V flag can be hooked up to be triggered by a hardware signal.You're not likely to use the V flag much.The V flag may be directly cleared by using instruction CLV (Clear overflow).If the V flag is clear, BVC (Branch overflow Clear) will branch; if the V flag is set, BVS (Branch overflow Set) will branch.

Hoisting the Flags

Consider the following code:

```
2ØØØ ADC #$Ø7     ;add 7 to the A register
2ØØ2 CPX #$Ø2     ;compare the contents of X
                  ;to 2
2ØØ4 LDY #$ØØ     ;load Ø into Y
2ØØ6 STA $2345    ;store the contents of A
                  ;into $2345
```

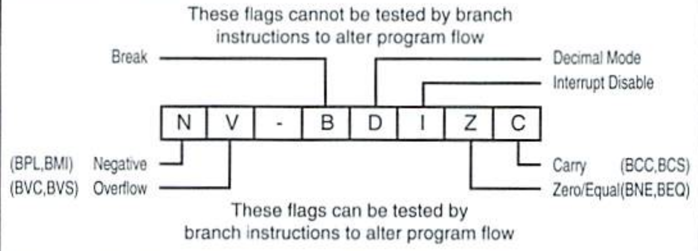

These flags cannot be tested by branch instructions to alter program flow
Break
Decimal Mode
Interrupt Disable
N
V
-
B
D
I
Z
C
(BPL,BMI) Negative
(BVC,BVS) Overflow
Carry (BCC,BCS)
Zero/Equal(BNE,BEQ)
These flags can be tested by branch instructions to alter program flow

Figure 1: The bits of the Processor Status Register (P) sometimes control the way the computer works. Four of the bits (N, V, Z, and C), or flags as they are often called, report the results of recent operations, and can be "tested" by the various branch instructions to alter program flow. For example, BCC (Branch if Carry Clear) will cause program operation to change to a new address if the C flag contains a zero.

Although we won't know the contents of the registers after this code has run (except register Y, which will contain zero), we can analyze how the conditional flags have been affected.

The first instruction, ADC, affected all four flags: N, Z, C, and V. But N, Z,and C will be changed by the next instruction. The second instruction, CPX, affected flags N, Z, and C. The Z flag will be set if the X register contains a value of two (an equals condition was found); the C flag will be set if the value in X is two or higher. N will be affected, but not in a generally useful way. However, the following instruction is going to modify the N and Z flags once more. The third instruction, LDY #$00, won't affect the V or C flags. But the Z flag will be set (we've placed a zero in the register) and the N flag will be clear (the high bit is not set).The last instruction, STA, affects no flags at all.

So our situation is this: V was last affected by the ADC instruction; C was influenced by the CPX; and the N and Z conditions are the result of the LDY instruction.

Quick Trick: Pushing the Status Register

We mentioned that the status register could be put away and brought back for checking at a later time. We do this with a PHP (Push Processor Status) to save to the stack, and PLP (Pull Processor Status) to bring it back.

Here's a useful application. If you are copying a file, say from disk to the printer, your program needs to check the input stream for an end-of-file condition. That condition will be logged in the status word ST, which is located at address hex 90 on the VIC-20 and subsequent machines. (On earlier machines, you'll find it at $96).

But the program needs to check the end-of-file condition AFTER the output is sent... at which time, the contents of ST have been changed by the output activity. We need to save the end-of-file indicator as it appears when we read a byte; but take action on its condition after the byte has been written. Do you see the "deferred decision"?

Here's the quick machine language code. Later, the BASIC program will place the code at address $2000 (not an ideal site, but almost universally available):

```
LOOP:
      LDX #1          ;connect input stream to
                      ;logical file 1 (which has
      JSR $FFC6       ;already been opened)
      JSR $FFE4       ;Read a byte into the A
                      ;register
      LDX $90         ;Read the status byte (at
                      ;this time status byte will
                      ;be Ø if not end of file ..
                      ;and thus, the Z flag would
                      ;be set)
      PHP             ;now, that Z flag is safe
                      ;on the stack
      PHA             ;let's put our input byte
                      ;there, too
      JSR $FFCC       ;disconnect input stream
      LDX #2
      JSR $FFC9       ;connect output stream to
                      ;logical file 2
      PLA             ;bring back the input data
                      ;byte
      JSR $FFD2       ;transmit it
      JSR $FFCC       ;disconnect output stream
      PLP             ;bring back the Z flag; if
      BEQ LOOP        ;it's set, get more
```

Here's some Basic code to do the whole job:

```
100 DATA 162,1,32,198,255,32,228,255
110 DATA 166,144,8,72,32,204,255
120 DATA 162,2,32,201,255,104,32,210,255
130 DATA 32,204,255,40,240,226,96
200 FOR J=8192 to 8222
210 READ X : T=T+X
220 POKE J,X
230 NEXT J
240 IF T<>4390 THEN STOP : REM DATA ERROR
300 OPEN 1,8,3,"DATAFILE"
310 OPEN 2,4 : REM OUTPUT TO PRINTER
320 SYS 8192
330 CLOSE 2
340 CLOSE 1
```

Put the name of a sequential file into line 300 (I've used "DATAFILE"). You may change line 210 to output wherever you desire: OPEN 2,3 would send to the screen, for example. If you have a pre-VIC-20 computer, you'll need to change the value 144 on line 110 (and modify the checksum value on line 240); but I'll leave that as an exercise.

TABLE 1: INSTRUCTIONS AFFECTING FLAGS

Affects Z, N, C, and V
- ADC - Add with Carry
- PLP - Pull Processor status register
- RTI - Return from Interrupt
- SBC - Subtract

Affects Z, N, and C
- ASL, ROL - Arithmetic shift/rotate left
- LSR, ROR - Logical shift/rotate right
- CMP, CPX, CPX - Compare A, X, Y

Affects Z and N
- AND, EOR, ORA - Logical AND, Xor, OR
- DEC, DEX, DEY - Decrement memory, X, Y
- NC, INX, INY - Incrememt memory, X, Y
- LDA, LDX, LDY - Load A, X, Y
- PLA - Pull A from stack
- TAX, TAY, TXA, TYA - Register transfers
- TSX - Transfer Stack pointer to X

Special
- BIT (N, V, Z) - Test bits at address
- CLC, SEC - Clear/Set Carry flag
- CLV - Clear overflow flag

Affects no conditional flags
- BCC BCS BEQ BNE BMI BPL BVS BVC - all Branches
- BRK - Break
- CLD, SED - Clear/Set decimal mode
- CLI, SEI - Clear/Set interrupt lockout
- JMP, JSR - Jump, Jump to subroutine
- NOP - No operation
- PHA, PHP - Push A, Push procesor status regiser to stack
- RTS - Return from Subroutine
- STA, STX, STY - Store A, X, Y
- TXS - Transfer X to stack pointer.

PERIPHERAL VISION

THE PERIPHERAL BUS

By Jim Butterfield

Before the VIC-20, Commodore computers talked to peripheral devices using a *parallel* bus called the *IEEE-488* or *GPIB*. Subsequently, this was replaced by a *serial* bus. The principles of the two were the same.

Some of what follows may be differ slightly if you're using some form of "fast bus", such as JiffyDOS. But the description will be generally valid.

About the Bus

A *bus* may be loosely defined as a collection of wires that connect several different devices. The Commodore bus design connects the computer with a number of peripheral devices, mostly disk drives and printers. The original (parallel) design was based on the GPIB (*General Purpose Interface Bus*) designed by Hewlett-Packard, later adopted as international standard IEEE-488.

Hewlett-Packard's objective was to allow instruments to be linked together, perhaps with a computer included. There was no central control: any device could be made master of the bus and command the others; the device might later give up control in favor of another. There were also polling modes defined, where the controller could send out a query, asking if any device had anything interesting to report.

The first Commodore PET and CBM computers kept faithfully to the Hewlett-Packard design. In fact, some users connected GPIB instruments to these early computers for process control and monitoring. Commodore deviated slightly in a few areas. The computer used an economical edge connector, rather than the more elaborate (and expensive) standard IEEE-488 connector. Commodore decided that the computer would be the only controller on the bus. Features such as polling were rarely used (an early Commodore acoustic modem was the only device that recognized polling). Commodore peripherals were simplified, but made less efficient, by having them lock out the bus when they were busy.

In the pioneer days of the first home computers, Commodore's bus design was sophisticated and brilliant. Over the years, though, the rest of the industry moved toward discrete connections: the Centronics interface for printers, RS-232 ports for modems, and dedicated controllers for disk drives. The switch to the serial bus slowed things down, and Commodore started to be left behind. But give our guys credit: in those first days when no standards existed, Commodore introduced a remarkable design with their peripheral bus and intelligent devices. And todays mainstream computer industry seems to be about to rediscover the concept of a peripheral bus.

How the Wires Work

This description applies to both the IEEE-488 and the serial bus. All wires connect to all devices. The bus has no "direction"—every device, in principle, can place a signal on a wire which will be seen by all devices. To be a little more technical: every signal wire on the bus is normally held *high* at around five volts (logic 0 in *negative logic*); but any device can pull the voltage *low* to ground (logic 1), and this changed level will be seen by all devices. In fact, several devices could be pulling the voltage low; only one is needed to do the job.

Because "everything connects to everything", a bus can be extended by means of Y connectors. Usually, the serial bus uses the two-connector set on disk drives to continue the hookup to other devices. The IEEE-488 bus uses a male/female combination connector that allows extra cables to be easily hooked in.

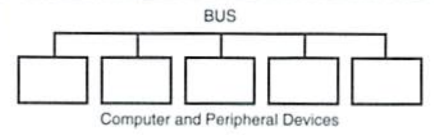


Figure 1: The computer and its peripherals are connected by a common bus. All wires connect to all devices; the computer could be anywhere in the chain.

Information (*data*) flows over the data wires: there are eight of these on the IEEE bus, but only one on the serial bus. To help the data move, there are additional control wires—often called *handshaking lines*—that help the timing of the data and acknowledge data receipt. These are carefully planned: only one device can send at a time, but several can be receiving data simultaneously.

This capability to broadcast data to several devices is important. It's especially needed for device selection, where the computer calls all devices and instructs one to stay connected, the others to drop off the line for the time being.

The *attention* (ATN) line is used for this selection. When the ATN line is pulled low, all devices listen to see if they will be involved in the following data transmission.

The Selection Mechanism

When ATN is set TRUE (low) by the computer, all other devices start to watch for incoming data signals. The computer sends one or more characters, which might mean such things as "device 4: listen" (one character); "device 8, secondary address 2: talk" (two characters); or, "device nine, secondary address 5: open file DATA for reading" (about six characters).

The major commands that are sent are TALK and LISTEN, which are bundled with the corresponding device number. The device number is specified in the lowest 5 bits (0 through 4) of a byte (see Figure 2a) and can contain a value from 4 to 30. While these five bits could actually hold a value of 0 through 31, the values 0 through 3 and 31 all have special meanings. Bits 5 and 6 of the same byte are used to indicate LISTEN and TALK, respectively.

-	T	L	Device Number (4-30)

Figure 2a: Device selection byte. The first byte sent after ATN has been pulled low. T controls TALK; L commands LISTEN; Device number 31 controls UNTALK and UNLISTEN.

Control	SA (0-15)

Figure 2b: Secondary address control byte. Optional; sent after the device selection byte. Control codes (in binary):

1111 - OPEN
1110 - CLOSE
0110 - data

To tell device 4 to listen, a value of $24 (hexadecimal) is sent. This presents a bit pattern of 00100100. Device 4 would be asked to TALK with a value of $44 (hex) although since that device is usually a printer, it's not likely to say much. The reverse commands, UNTALK and

UNLISTEN, typically carry a device number of 31, which means that UNLISTEN sends hex $3F, and UNTALK sends hex $5F.

The TALK or LISTEN command may be followed by a secondary address signal. This byte will carry not only the secondary address (from 0 to 15), but the high nybble will signal whether this transmission will open a file, close a file, or transport data. OPEN is signaled by a high nybble of $F (hex), CLOSE by a high nybble of $E, and other data activities by a high nybble of $6. So, to ask device 8, secondary address 3 to send more information from an already open file, the computer would send two bytes while ATN is pulled low: $48 and $63 (hex) . The first byte ($48) would mean "talk, device 8", and the second ($63) would be "data, secondary address 3".

A file name may be sent if the operation is an OPEN, but that will happen after ATN is turned off, so the information classifies as regular data.

The Byte Exchange

The mechanism is the same whether or not the ATN mode is active. The bits are transmitted eight at a time, over eight wires, on the IEEE-488 bus. On the serial bus, the bits go over a single wire, one at a time, with a "clock" line signaling that each bit is ready. After a byte is sent, the sender waits for an acknowledgement.

The acknowledgement system has to be planned carefully, since there may be more than one receiver. On the IEEE bus, the acknowledgement is sent with a signal on the NDAC ("Data Not [yet] Accepted") line. As long as any listening device holds this connection in the TRUE (low) state, the byte transmission will be considered as "not complete". With the serial bus, it's done in a trickier way: the receiving device pulls the data line TRUE (low) to acknowledge that a byte has been received; later, when all receiving devices have released the data line to FALSE (high), the next byte can be transmitted.

There needs to be a way for the talker to signal, "This is the end of the data". The signal is most often used to indicate the end of an incoming file. On the IEEE-488 bus, a separate line (EOI, for "End or Indicator") is used. On the serial bus, the condition is signaled by an additional handshake before the last byte is sent.

Tracking Bus Events

Let's try to follow the actions on the bus when the following three Basic statements are executed:

```
OPEN 1,4
PRINT#1,"HELLO"
CLOSE 1
```

OPEN 1,4: The computer sends NOTHING to the serial bus. If the Basic statement had contained a secondary address or a file name, then the bus would have been opened and commands and data sent. As it is, the computer simply makes a note in its internal file tables.

PRINT#1,"HELLO": This takes place in three steps. First, the computer signals "Device 4, listen" by turning ATN on and sending one byte, hex 24; then ATN is turned off. Device 4 is now in the "listen" mode, while all other connected devices are deaf. The second step is to send the "HELLO" data: the characters go out to the bus one at a time. Because of the way BASIC works, a RETURN character will also be sent. Finally, the ATN line goes on again, and an UNTALK signal is sent to the bus. All devices are effectively offline now.

CLOSE 1: The computer notes that logical file number 1 does not have a secondary address associated with it. NOTHING goes to the serial bus, but the file entry for logical file 1 is scrapped. If the file did have a secondary address, the device would have been selected and the CLOSE signal sent.

That CMD Command

No, we're not talking about the company, but about the BASIC command CMD. Here's what CMD does: it sends a LISTEN to the device concerned, and then leaves the bus wide open. At the same time, anything that would normally be directed to the screen will be redirected to the bus.

So if we want to get a hard copy of a BASIC program, we might type the sequence:

```
OPEN 4,4 : CMD 4 : LIST : PRINT#4 : CLOSE4
```

A quick rundown on what the commands do:

OPEN 4,4: as before, nothing goes to the bus, but an entry is made in the computer's "active file" list.

CMD 4: LISTEN is sent to device 4, and the bus is left open. Any output that normally would go to the screen will be redirected to the bus. The printer just sits there and listens for data.

LIST: The program listing would normally come to the screen. But since it's redirected, it flows out to the open bus, where it will hopefully be printed. The bus remains open.

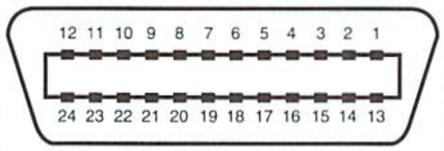
PRINT#4: The three steps: send LISTEN; send a RETURN character; send UNLISTEN. That last item is the one we want: it closes the bus, and restores output to the screen.

CLOSE 4: No bus activity. The file is removed from the computer's "active file" list.

Conclusion

The two bus systems, although mechanically different, use the same logic to communicate between computer and device. At some future time, we might go into details of the serial bus signals and timing.

IEEE-488 Connector



PET/CBM IEEE-488 Port

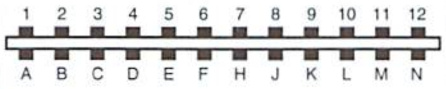


TABLE 1A: IEEE-488 Bus Pinouts

1	1	DIO1	Data Input/Output Line#1
2	2	DIO2	Data Input/Output Line#2
3	3	DIO3	Data Input/Output Line#3
4	4	DIO4	Data Input/Output Line#4
5	5	EOI	End or Identity
6	6	DAV	Data Valid
7	7	NRFD	Not Ready For Data
8	8	NDAC	Data Not Accepted
9	9	IFC	Interface Clear
10	10	SRQ	Service Request
11	11	ATN	Attention
12	12	GND	Chassis ground (cable shield)
A	13	DIO5	Data Input/Output Line#5
B	14	DIO6	Data Input/Output Line#6
C	15	DIO7	Data Input/Output Line#7
D	16	DIO8	Data Input/Output Line#8
E	17	REN	Remote Enable
F	18	GND	DAV Ground
H	19	GND	NRFD Ground
J	20	GND	NDAC Ground
K	21	GND	IFC Ground
L	22	GND	SRQ Ground
M	23	GND	ATN Ground
N	24	GND	Data Ground

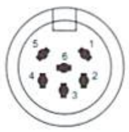


TABLE 1B: Serial Bus Pinouts

1	SRQ	Serial SRQ in
2	GND	System Ground
3	ATN	Serial ATN (attention)
4	CLK	Serial data clock
5	DATA	Serial data
6	RESET	Resets all devices

BASIC INSTINCTS

DYNAMIC FRACTALS

Program listings provided in this article are for the C-64 or C-128 in 64 mode.

By David Pankhurst

Last issue we looked at fractals from the point of view of self-similarity. Triangles within triangles, leaves within leaves. This month, we continue our look at fractal objects, but in the area of dynamic systems. Although there are a few definitions for the word dynamic, the one we are concerned with here refers to a continuously changing system. An asteroid in the far reaches of space is not a good example of a dynamic system, but a pendulum is. One simple example of a dynamic system is found on your calculator. Punch in the number 1.1, and press the squaring key. Eventually, it overflows, but not before you see a wild progression of numbers, each depending on the previous input, which was the result of the input before.

Dynamic modeling and equations are a means of analyzing the real world. If something exhibited X*X growth, pressing the square key on a calculator would model it exactly, and allow someone to make forecasts. But as in the old joke 'you can't get there from here', there is no direct path for answers. To estimate what the answer is after the seventh keypress, you have to enter the 1.1, and press seven times.This is an example of sensitivity to input. Looking at the example of the calculator again, I can enter 1.1, and press the square key. Along the way to overflow, I notice the number 2.14358881, which I note down. Later, in repeating the sequence, I figure I'll save time and a few keypresses by just entering 2.14 instead of 1.1. The result? By the time I overflow, the difference between this new pattern and the older one is 65%. Rather than an error of 3/1000 remaining small, it grew surprisingly large. Even if I entered the full number 2.14358881, there would be a variation eventually (the display is still not exactly the same as the number stored in the calculator). This extreme sensitivity to input means I can't repeat the sequence exactly without starting from the beginning.

This was the problem experienced by Edward Lorenz, a man who later went on to work on Chaos theory. When he was a meteorologist, he felt it was possible to predict the weather with equations. After coming up with a model that he felt satisfactorily reflected the weather, he then proceeded to rerun it from a later point in the simulation (as in the example with the calculator). Although the numbers were only off by thousandths from the computer's numbers, the small error quickly changed the results wildly. It was this problem of small effects having a large influence that is reflected in the phrase 'butterfly effect'—since small effects make such a difference in the weather (as well as other dynamic systems) it might well be that the flap of a butterfly's wings could start a hurricane.

This same sensitivity to input is at the core of fractal art. Two adjacent points on the screen could be different colors, representing different results; the points may differ by only small fractions, but the dynamic equations they represent are sensitive to input, leading to the differences. There's no equation we can use to enter "Thursday the 10th of March" and get back "sunny, mild", but with the C64, we can explore the dynamic equations that form the basis of much of the fractal artwork you see today.

Complex Ways of Looking at Things

When I press the square key, the number 1.1 moves toward infinity. Likewise, all numbers above one do the same (and for negative numbers, those less than minus one). And between one and minus one, all numbers move toward zero. We can get a visual feel for this behavior by plotting the results on a line. For every point we test, we mark it black if it goes to infinity, red if it goes toward zero, and white if it heads nowhere. (one and minus one are the only examples of the latter). The line we draw would soon be a length of black, with two white dots, and a piece of red in the center. Although not very interesting, it does show with one picture the dynamic system that is our calculator's squaring key.

But how do we get all those pretty pictures in books on fractals? They obviously don't use a line. What they do is model different dynamic systems from different views. But the one thing they have in common is their use of the complex plane to plot results.

For those who are in need of a refresher, the complex plane is a two-dimensional representation of complex numbers. And what are complex numbers? They are numbers that contain both a real and an imaginary part. Imaginary is the unfortunate term applied to numbers that are multiples of the square root of negative one. Since there is no real-world example of a square root of a negative number, 'imaginary' seemed to fit. Now however, I'm sure it only serves to make mathematics more confusing.

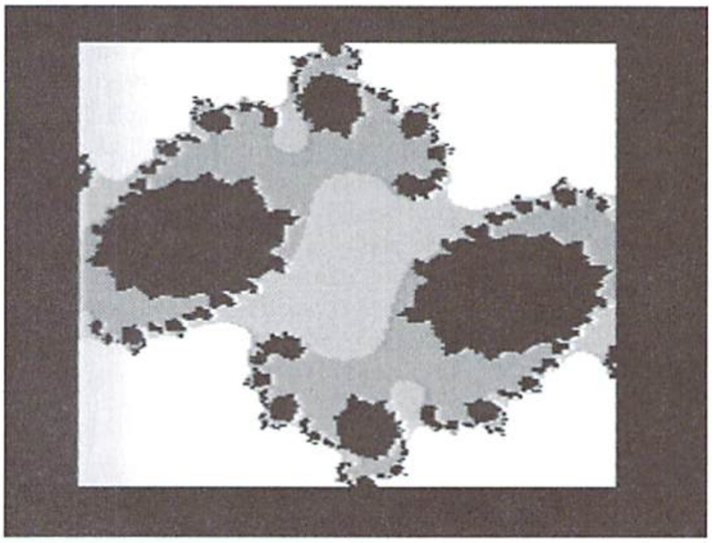
The rules for imaginary numbers are simple—let the letter i represent the square root of minus one. Then all imaginary numbers are multiples of this value. The square root of -16 is 4i, -25 is 5i, and so on. Multiplying two imaginary numbers results in the i's being multiplied as well, leaving minus one; 20i*15i=-300, and 3i*7i=-21.

Real numbers and imaginary numbers are linked in mathematics. A real number can have an imaginary part, and vice versa. Because i cannot be reduced further and combined with the real part, they are joined with a plus symbol:

2.34+5i
0+2i
15.4+0i

As in the examples above, if a number is missing an imaginary or a real part, that coefficient is zero.

The C-64 does fractals using FRACDRAW.BAS ▶



Displaying these numbers is where we get the graphs for fractals. By convention, the imaginary part is along the Y axis, with the real part plotted along the X axis. The result is that a number gains a two-dimensional quality. For example, 5+2i would be a point placed at X=5 and Y=2. Likewise, the point at X=3, Y=4 would be the complex number 3+4i.

Julia and His Sets

Now we have the tools to discuss fractals. In the first quarter of the century, a mathematician named Gaston Julia was examining the dynamics of complex numbers. Unlike our little graph of the calculator square key squaring complex numbers is much more interesting. Take for example:

```
 2 + 3i
X7 + 8i
```

As in polynomial math, the result would be each part of the top multiplied by each part of the bottom:

```
(2+3i)(7+8i)=(2*7)+(2*8i)+(3i*7)+(3i*8i)=14+16i+21i+24i*i
```

The i*i resulted in minus one, leaving :

```
14+16i+21i+24*i*i=14+37i-24=-10+37i
```

Working with complex numbers, Julia eventually settled on a rule that would result in an interesting system:

1) take a number from the complex plane, z, and square it
2) add a complex number constant c to the new value of z
3) repeat steps one and two with the new value of z, until you're sure it's going toward infinity or not

Step three, testing for infinity, isn't as awkward as it sounds. If a complex number's absolute value goes over two, it is destined for greatness. This solves one problem, but adds another: what is a complex number's absolute value? It is the distance from the origin (complex number 0+0i) to the point in question. Looking at a graph, it would be the hypotenuse of a right sided triangle of height Y (the imaginary part) and length X (the real part), or the square root of (Y*Y+X*X).

With all these details in place, let's look at the code fragment given in FRAGMENT.BAS. This routine squares a complex number and adds a constant. It repeats this up to 31 times (line 130) or until the absolute value is greater than two.If it passes this limit, a color is plotted on screen at that point, representing how long it took to fly off. Typically, black is for the quick flyers, and white is reserved for those that never fly away (at least after testing here 31 times).

Lines 90 and 100 do the squaring. From the earlier example, multiplying each complex number results in four multiplications. Adding the constant adds another operation. These lines are a simplification of the product of:

```
(ZREAL+ZIMAG*i)*(ZREAL+ZIMAG*i)+(CREAL+CIMAG)
```

In displaying fractals, two colors are rarely used—much of the interest comes from the boundary, as numbers more quickly head toward infinity. Using COUNT in the example above, up to 31 different shades could be used; for the C64, only two (high resolution) or four (medium resolution) are available, so scaling is done.

```
√Σ     FRAGMENT.BAS

206    13 rem fractal output
3      2Ø zreal=.3 :zimag=.5 :rangex=.2:rangey=
       .2 : creal= Ø.435 : cimag=Ø.45
2ØØ    3Ø for x=Ø to 159 :rem plot on screen
231    5Ø for y=Ø to 199
71     6Ø zreal=x/16Ø*2*rangex-rangex :rem get
       real part of point
129    7Ø zimag=(1-y/2ØØ)*2*rangey-rangey :rem
       get imaginary part of point
13Ø    8Ø count=Ø
238    9Ø tempa=zreal*zreal-zimag*zimag+creal :
       rem square z and add c
1ØØ    1ØØ tempb=2*zreal*zimag+cimag
251    11Ø zreal=tempa:zimag=tempb :rem put res
       ult into z
23Ø    12Ø count=count+1
1ØØ    125 rem test z if heading for infinity &
        check if count done
139    13Ø if sqr(zreal*zreal+zimag*zimag)<2 an
       d count<31 then 9Ø
41     14Ø rem here plot color to x,y based on
       size of count
17     15Ø next : next
```

Viewing Gaston's Discovery

The final program for fractal drawing is FRACDRAW.BAS. Medium resolution is used, giving four colors and a better look to the output. The program also makes heavy use of machine language for speed. With it, a full screen fractal can be displayed in under four hours, depending on complexity. [*Note: FRACDRAW.BAS has been modified to add optimization for SuperCPU users, reducing drawing time down to about half an hour. These changes will not affect computers that are not SuperCPU-equipped. -Ed.*]

Here are some notes concerning the program. Firstly, lines 135 and 140 cannot be moved. To simplify the machine language, I had the routine grab information from the first five variables in memory. The CLR at line 135 guarantees the variables at line 140 are the first to be initialized, but if you insert other variables, or move the lines, it won't work.

The black background is set at line 115 (POKE 53281,0). The variables C1 and C2 at line 145 are the middle shades of gray, and the value poked to 54272+I is the white color. Feel free to adjust them.

Line 155 is where the complex constant (CR and CI) is initialized, as well as the range (DX and DY). The range represents how far off-axis the view is: 0.8 means the display stretches from X=0.4 to X=-0.4, with the same for Y.

Try the program as it is, and you will get a nice fractal display in under four hours. Depending on the fractal, other displays will be quicker. Modifying the program can involve changing the constant (CR and CI), the loop (more loops mean sharper detail, but more time plotting), and size (try limiting X and Y to a smaller portion of the screen, or use STEP 2 in line 160 to plot only every second point as a fast way of looking at a fractal). Also, the viewing range can be adjusted with RX and RY. Smaller values will show greater detail, but don't set it above RX=4 or RY=4. (can you guess why?)

Some examples of CR and CI values to try are:

CR	CI
+0.300	0.500
-0.122	0.745
-0.754	0.049
+0.354	0.536
-0.744	0.097
-0.756	0.097
-0.756	0.297
+0.736	0.097
+0.766	0.097

The program, by the way, plots what is called the Julia set—the set of complex numbers that will grow increasingly large as they are repeatedly squared and a constant added. The constant makes all the difference. If you have the time, set the constant to zero and see what happens (see if you can predict it in advance). The white section represents the'prisoner set'—numbers that will not go toward infinity as you perform the calculation. The other colors form the 'escape set' (for obvious reasons) with the colors indicating the speed they head toward infinity.

For the Julia sets, CR+CI is set by you, and ZR+ZI is initially set to the point being plotted. But what would happen if you were to set CR+CI to the point being plotted, and ZR+ZI to zero? Then the plot would be effectively the escape set for all Julia sets, for every constant input. This result is the familiar Mandelbrot set. To see it, add the line:

```
163 CR=ZR:CI=ZI:ZR=Ø:ZI=Ø
```

Summary

The Mandelbrot set and Julia sets are just a few examples of dynamic systems on the complex plane. In each case, complex numbers lend themselves to interesting and striking displays. But are they fractals? Recently, mathematicians proved that at least the Mandelbrot set is. Even though it can look like there are pieces disconnected from the main mass, in reality everything is one object, connected by filaments, with of course the fractal's extremely high perimeter length (probably infinite, but I'm not sure if that's been mathematically proven).

Fractals of all sorts provide an insight into life that we normally don't explore. One picture shows the dynamics of a whole system, and enlarging the detail only adds to the richness. That there is beauty in what we see is, I think, the most interesting aspect of all. Esthetics are determined by a number of factors. Are we programmed in some way to appreciated fractal geometry? Is it familiar in the day-to-day world, and so we find the familiar comfortable and pleasant? In any case, with these programs, you can explore the fascinating world of fractals, and decide for yourself.

```
√Σ   FRACDRAW.BAS

151  1ØØ rem julia set fractal display
245  1Ø2 rem
228  1Ø5 print"poking":fori=49152to49522:read
     x:pokei,x:c=x+c:next:ifc<>34343thenstop
1Ø5  11Ø rem set up medium res screen
223  115 poke 5328Ø,7:poke 53281,Ø
29   12Ø poke 5327Ø,peek(5327Ø)or16
74   125 poke 53265,peek(53265)or32
59   13Ø poke 53272,peek(53272)or8
129  135 print"{CLEAR/HOME}";:dim i(3ØØØ):clr
8Ø   14Ø zr=Ø:zi=Ø:cr=Ø:ci=Ø:n=5:f=6:m=49152
22Ø  145 c1=12:c2=15:fori=1Ø24to2Ø23:poke5427
     2+i,1:pokei,c1*16+c2:next:rem do colors
9Ø   15Ø poke 5328Ø,Ø:rem signal-ready
63   155 cr=-Ø.756:ci=Ø.197:dx=.8:dy=.8
151  16Ø for x=Ø to 159:for y=Ø to 199:zr=x/8
     Ø*dx-dx:zi=(2-y/1ØØ)*dy-dy:n=32
187  162 poke53367,Ø:rem supercpu optimizatio
     n off
186  165 sys m,x,y,f/8:if peek(198)then x=1e9
     :y=x
172  167 poke53366,Ø:rem supercpu optimizatio
     n on
161  17Ø next:next:poke53367,Ø:rem supercpu o
     ptimization off
98   175 wait 198,7:getx$
191  18Ø print"{CLEAR/HOME}":poke53265,27:pok
     e 53272,21:poke5327Ø,2ØØ
211  53Ø data 76,14Ø,192,76,14,192,76,12Ø,192
     ,76,128,192,234,234,32,6,192,133,4
13Ø  54Ø data 32,6,192,133,5,74,74,74,72,133,
     3,74,74,24,1Ø1,3,133,3,1Ø4,1Ø,1Ø,1Ø
189  55Ø data 1Ø,1Ø,1Ø,133,2,24,165,5,41,7,1Ø
     1,2,133,2,144,2,23Ø,3,165,4,41,252
24Ø  56Ø data 72,24,1Ø1,2,133,2,144,2,23Ø,3,1
     Ø4,24,1Ø1,2,133,2,144,2,23Ø,3,24,169
2    57Ø data 32,1Ø1,3,133,3,32,6,192,41,3,72
     ,165,4,41,3,73,3,17Ø,1Ø4,224,Ø,24Ø
111  58Ø data 5,1Ø,1Ø,2Ø2,2Ø8,251,16Ø,Ø,17,2,
     145,2,96,32,253,174,32,158,183,138
4Ø   59Ø data 96,24,1Ø1,45,72,165,46,1Ø5,Ø,16
     8,1Ø4,17Ø,96,169,3Ø,32,9,192,32,162
4Ø   6ØØ data 187,32,155,188,165,1Ø1,133,2,16
     9,2,32,9,192,32,162,187,169,2,32,9
51   61Ø data 192,32,4Ø,186,169,37,32,9,192,3
     2,212,187,169,9,32,9,192,32,162,187
248  62Ø data 169,9,32,9,192,32,4Ø,186,32,18Ø
     ,191,169,37,32,9,192,32,1Ø3,184,169
8Ø   63Ø data 16,32,9,192,32,1Ø3,184,169,37,3
     2,9,192,32,212,187,169,2,32,6Ø,188
7    64Ø data 169,9,32,9,192,32,4Ø,186,169,2,
     32,9,192,32,4Ø,186,169,23,32,9,192
238  65Ø data 32,1Ø3,184,169,9,32,9,192,32,21
     2,187,169,37,32,9,192,32,162,187,169
146  66Ø data 2,32,9,192,32,212,187,169,2,32,
     9,192,32,162,187,169,2,32,9,192,32
2Ø3  67Ø data 4Ø,186,169,37,32,9,192,32,212,1
     87,169,9,32,9,192,32,162,187,169,9
196  68Ø data 32,9,192,32,4Ø,186,169,37,32,9,
     192,32,1Ø3,184,169,37,32,9,192,32,212
1ØØ  69Ø data 187,169,4,32,6Ø,188,169,37,32,9
     ,192,32,91,188,2Ø1,1,2Ø8,7,198,2,24Ø
185  7ØØ data 3,76,155,192,165,2,32,6Ø,188,16
     9,37,32,9,192,32,212,187,76,3,192
```

always **SAVE**

before you **RUN**

CHECKSUM

Commodore World's Program Entry Checking Program and Tips on Entering Programs from this Magazine

CHECKSUM is a program that proofreads your typing when you enter a listing from the magazine. It assigns a numerical value to each character that you type, adds up the values of the line you typed and displays the sum. (Checksum, therefore, means that it checks your typing by summing the characters.) It also verifies that you have typed the characters in the proper order. (Checksum won't tell you if you miss a line of code entirely, so verify that yourself.) Checksum runs "in the background" when you type in lines of program code. Whenever you type a line and press RETURN, Checksum will display a value. Compare that value to the value published next to the line of code in the magazine. If the numbers match, you've typed the line correctly. Simple.

Typing in CHECKSUM

First, type in Checksum carefully from the listing on this page. Be sure to press RETURN after every line to enter it into memory. Once you have typed the program, *save it*. In fact, save it a few times while you're typing, just to be safe. (This is good advice whenever you type in a program. I usually change the name each time I save; for example, Checksuml, Checksum2, and so on.) Double-check your work, making sure that you've typed in every line and that you've pressed RETURN after every line you've typed. If you make errors when typing in Checksum, a test run of Checksum will tell you which line is incorrect. (This safety feature works only in the Checksum program itself, and does not apply to any other listings in the magazine.) Whenever you find a typing error (in any program listing), fix it, press RETURN to enter the change, save the program again and try another run. Repeat this process as often as necessary. Important tip: Don't get discouraged if the program won't run. Be patient. Be thorough. It will work eventually. You'll know your Checksum is ready when you see the line:

```
    TO TOGGLE ON OR OFF, SYS XXXX
```

Entering Programs Using CHECKSUM

When you're ready to type in your first listing from the magazine, load and run Checksum. Make a note of the number that is displayed on the screen (49152 for the C-64; 3328 for the C-128). To activate and deactivate Checksum, type SYS followed by that number, then press RETURN. You need to have Checksum active whenever you're typing in a listing. Checksum must be deactivated, however, when you run the new program. The next step is typing in a new program listing as it appears in the magazine.

As you begin, you'll notice that to the left of the start of each line is a number. Don't type this number in: It's simply the Checksum value. Stop typing at the end of the program line and press RETURN. If you've typed the line correctly, the number displayed on the screen will match the Checksum value. If the numbers don't match, you've made a mistake. Check the line carefully, make your changes and press RETURN. The computer won't know you've made a change unless you press RETURN on the changed line to enter it. A few type-in hints: The Checksum does not verify blank spaces in the program lines unless they are within quotation marks, because adding or omitting such spaces will not affect the operation of the program. The exception to this is hexadecimal Data statements. These are the Data statements, such as this one, that don't have commas:

```
    100 DATA 1234567890123456789O*123456789012345
    67890*12345678901234567890*
```

In statements such as these, you must have one space between the word DATA and the numbers that follow. Checksum will not catch that error.

Special Key Combinations

As you type, you may be confused the first time you see curly braces {}. These braces mean "perform the function explained within." For example, {22 SPACES} means that you need to press the space bar 22 times. Don't type the braces (you can't, of course, because there are no curly braces in the Commodore character set). Here are some other common examples:

{CLEAR/HOME}	hold down the SHIFT key and press the CLR-HOME key.
{2 CRSR DN}	tap the cursor down key twice.
{CTRL i}	hold the CONTOL key and press the I key.
{CMDR t}	hold down the COMMODORE key and press the T key.

Continue typing in your program, saving often and checking each checksum value with the one in the magazine, until you've finished the listing. Phew! So now you're ready to run your program, right? Not quite. First, save it. Second, deactivate Checksum by typing SYS followed by 49152 for the C-64 or 3328 for the C-128. Now you can run. Don't be discouraged if you still get an error. It happens. Use Checksum faithfully. Be patient. Be thorough. It will work eventually.

CHECKSUM

```
100 rem cw checksum 64/128
110 mo=128:sa=3328
120 if peek(65533)<>255 then mo=64:sa=49152
130 i=0:ck=0:ch=0:ln=300
140 for k=0 to 16
150 for j=1 to 10
160 read b:if b>255 then goto 280
170 ch=ch+b:poke sa+i,b:i=i+1
180 next j
190 read lc:if lc<>ch then goto 280
200 ch=0:ln=ln+10
210 next k
220 pokesa+110,240:pokesa+111,38:pokesa+140,234
230 printchr$(147):print"cw checksum";str$(mo):print
240 print"to toggle on or off, sys";sa:if mo=128 then 270
250 pokesa+13,124:pokesa+15,165:pokesa+25,124:pokesa+26,165
260 pokesa+39,20:pokesa+41,21:pokesa+123,205:pokesa+124,189
270 pokesa+4,int(sa/256):sys sa:new
280 print"you have a data error in line";ln;"!":end
290 rem do not change these data statements!
300 data 120,162,24,160,13,173,4,3,201,24,884
310 data 208,4,162,13,160,67,142,4,3,140,903
320 data 5,3,88,96,32,13,67,152,72,169,697
330 data 0,141,0,255,133,176,133,180,166,22,1206
340 data 164,23,134,167,132,168,170,189,0,2,1149
350 data 240,58,201,48,144,7,201,58,176,3,1136
360 data 232,208,240,189,0,2,240,42,201,32,1386
370 data 208,4,164,180,240,31,201,34,208,6,1276
380 data 165,180,73,1,133,180,230,176,164,176,1478
390 data 165,167,24,125,0,2,133,167,165,168,1116
400 data 105,0,133,168,136,208,239,232,208,209,1638
410 data 169,42,32,210,255,165,167,69,168,170,1447
420 data 169,0,32,50,142,169,32,32,210,255,1091
430 data 32,210,255,169,13,32,210,255,104,168,1448
440 data 96,104,170,24,32,240,255,104,168,96,1289
450 data 56,32,240,255,138,72,152,72,24,162,1203
460 data 0,160,0,32,240,255,169,18,208,198,1280
```

User Group Connection

Support your local User Group

Looking for a Commodore user group in your area? User groups can help you solve problems, keep you informed of new products and events that might be of interest, and give you a chance to share your computing experiences with others who enjoy Commodore computing. **The following are confirmed groups and verified addresses, registered with *Commodore World*.**

ALABAMA

H.A.C.K.S., 9408 Lynn's Terrace, Huntsville, AL 35802

CALIFORNIA

South Bay Computer Groups, PO Box 189 ,Chula Vista, CA 91912-1899

Southeast San Diego C-64 User's Group, 9830 Dale Ave. #24, Spring Valley, CA 91977-2445

Commodore Technical User's Group, 2231 E. Trenton Ave., Orange, CA 92667-4451

Civic 64/128, PO Box 2442, Oxnard, CA 93034

COLORADO

Colorado Commodore Computer Club, 12246 Monroe Place, Denver, CO 80241

FLORIDA

Fort Walton User Group, 221 Baker St, Fort Walton Beach, FL 32548

Lake/Sumter C.U.G., PO Box 416 , Leesburg, FL 34748

GEORGIA

Stone Mountain User's Group, 703 Waldan Walk Cir., Stone Mountain, GA 30088

ILLINOIS

Illinois Commodore Users, P.O. Box 781, McHenry, IL 60050

Sandwich Computer User Group, PO Box 23, Sandwich, IL 60548-0023

Commodore Club of Rockford, PO Box 6341, Rockford, IL 61125-1341

MARYLAND

Meeting 64/128 Users Through the Mail, 4427 39th Street, Brentwood, MD 20722-1022

Gaithersburg Commodore User's Group, P.O. Box 5712, Durwood, MD 20855-0712

N.I.S.T, 3124 Pheasant Run, Ijamsville, MD 21754

NEBRASKA

Greater Omaha Commodore User's Group, PO Box 241155, Omaha, NE 68124-5155

TEXAS

Crestview Computer Club, 401-A Northern Dove Ln. ,Coperas Cove, TX 76522-8432

VIRGINIA

NOVACOM, 9206 Annhurst Street , Fairfax, VA 22031-1902

Washington Area CUG, 7728 Viceroy Street, Springfield, VA 22151

VERMONT

Green Mountain Commodore User's Group, PO Box 6087, Rutland, VT 05702-6087

WASHINGTON

Island Commodore User Group, 1675 N. Rientjes Ln ,Oak Harbor, WA 98277

Vancouver CUG, 1903 E. 9th St. ,Vancouver, WA 98661

WISCONSIN

Milwaukee Area Commodore Enthusiasts, PO Box 26216, Milwaukee, WI 53226-6216

Computer Users Support Group, PO Box 085682, Racine, WI 53408-5682

CANADA

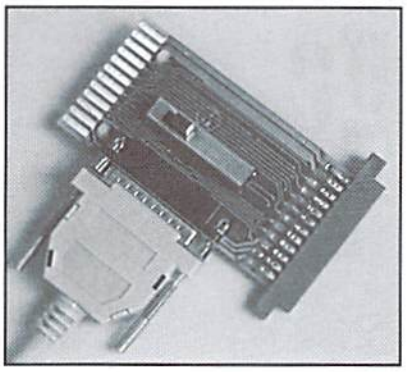
TPUG, 3605 Lakeshore Blvd. West, Box 48565, Etobicoke, Ontario, Canada M8W 4Y6

NEW ZEALAND

Christchurch Commodore Users' Group, P.O. Box 4665, Christchurch, New Zealand

GEOCABLE II

Parallel Print Cable



The Most Advanced Parallel Print Cable Ever!

- **Uses Standard Cable**
  Equipped with a female DB-25 cable connector to allow use of standard 'PC' printer cables.
- **Switchable Pass-thru**
  Allows connection of printer and other user port peripherals such as modems without conflicts.
- **Faster Output**
  Parallel printing offers up to a 40% increase in output speed.
- **GEOS Drivers Included**
  Includes GEOS drivers plus useful GEOS utilities like MacAttack II, WrongIsWrite, and GEOS CONVERT.

GEOCABLE-II (6 ft.)	**ONLY $29.00**
GEOCABLE-II (15 ft.)	**ONLY $34.00**

Shipping: US $5.00, Canada $7.00, Foreign $25.00.

GeoCable II is compatible with GEOS, Perfect Print LQ, Action-Replay MK VI, Superbase, Superscript, Paperclip III, and all software that supports user port parallel printing.

CMD **Creative Micro Designs, Inc.**
P.O. Box 646
East Longmeadow, MA 01028
Orders: 800-638-3263
Info: 413-525-0023

E X P A N D YOUR SYSTEM

3 SLOT CARTRIDGE PORT EXPANDERS

The EX2+1 and EX3 Cartridge Port Expanders bring new expandability to you Commodore 64 or 128. Combine the use of compatible cartridges. Disable cartridges not in use. Change the address your computer finds a cartridge at. Avoid the constant changing of cartridges that puts extra wear on your cartridge port. The EX3 offers 3 vertical expansion slots, while the EX2+1 provides 2 vertical slots and 1 horizontal slot. Both units offer 7 switchable signals per slot, address mapping on slot 2, and a reset button.

CALL 1-800-638-3263

EX3

$29.95 Plus S/H

EX21

$34.95 Plus S/H

Creative Micro Designs, Inc., P.O. Box 646, East Longmeadow MA 01028

Need Input? It Doesn't Get Any Better Than This...

$69.95 Plus S/H

Smart TRACK

For years, Commodore set the standard with the 1351 Mouse. Sure, it was vastly superior to using the old digital input devices like the 1350 mouse, joysticks or imitation trackballs. But everything can be improved. Guaranteed 100% 1351-compatible, SmartMouse and SmartTrack do everything the C-1351 does and more! These highly intelligent, three-button input devices include a built-in battery-backed Real-Time Clock, along with double-click and Turbo features for GEOS. Plus, they come with a complete set of utilities for using the clock in GEOS and BASIC applications. Modern ergonomic designs make these devices smooth operators that are a pleasure to use. So, if you're tired of the slow, erratic movement of your current input device, make the intelligent choice and pick up a SmartMouse or SmartTrack today!

$49.95 Plus S/H

Smart MOUSE

SmartMouse and SmartTrack... the Best C-64/128 Input Devices Ever!

- **Three buttons means convenience! If you're a GEOS user, the left button is configured as single click, the right as a handy double click and the center button is the TURBO button. When depressed, it doubles the speed at which the pointer moves across the screen. Additionally, programmers can assign their own functions to all three of the buttons.**
- **Unlike other third party mice or trackballs, the CMD SmartMouse and SmartTrack trackball use the same custom gate array chip as the Commodore C-1351 mouse to guarantee 100% compatibility.**
- **Switches easily into joystick emulation mode on power-up by holding down the right button.**
- **SmartMouse/SmartTrack utilize the same advanced technology used in today's powerful 486 and Pentium PC's, providing you with unparalleled accuracy and smoothness.**
- **Built-in battery-backed Real-Time Clock automatically sets the GEOS clock, displays time and can be used in your own programs.**
- **Includes utilities disk and detailed manual explaining the utilities and programming information.**
- **Attention Lefties! SmartMouse can be altered for left handed use.**
- **Don't be fooled by the old style digital "trackballs". These only perform as well as a joystick. SmartTrack uses analog inputs for greater accuracy.**

Over The Edge...

By Harold Stevens, Jr.

AHOY THERE, NO PIRATES ALLOWED!

With the absence of commercial software for the Commodore 64, piracy is on the rampage again, particularly when it comes to games, utilities and GEOS. This is something that no computer user should tolerate as it is both illegal and immoral.

Several years ago, I got into a heated discussion with a couple of users on a local Commodore board who were boasting about their ability to pirate software. These people justified their actions on the fact that there is no new commercial software available in this country and they have every right to "steal the warez" that continue to be produced in Europe and abroad, as commercial producers are ignoring the large followers of Commodore users that continue to flourish in North America.

If my memory serves me correctly, these were also the same people who did the same thing when commercially produced software was being distributed at the height of Commodore 64's popularity back in the mid-to-late 1980s. At that time their excuse was that the cost of software was too high and they had the right to "steal the warez" and distribute it to those who owned computers, but could not afford to shell out the money to support their hobby. Many of them fashioned themselves as modern day Robin Hoods and devoted computer bulletin board systems exclusively to pirating.

When I pointed out how wrong their actions were, one or two of these hackers became upset and flamed me by calling me "lame" or a couple of other cyberpunk terms that is not worth repeating. It appeared that they took the attitude that they were doing nothing wrong.

One thing I have discovered about those who hack software is that they are usually young and have a distorted sense of morality. Usually they don't get caught until they start serious hacking into the computer files of government offices, corporations or computer bulletin board systems. There were instances of a couple of high school aged kids in the Columbus, Ohio, area who were arrested by local and federal law enforcement officers because of their hacking. And these kids were doing their hacking with C-128 computers.

Many of these people don't realize it, but software piracy is among the top reasons that commercial software producers pulled themselves out of the Commodore market. How could these producers make money, when all people had to do to get their software was to log on to a pirate board and download a game or application that was zipped, lynxed or arced into a single file. For those who didn't want to pay out money to get software,this was a convenient way to get something for nothing.

This problem will probably never go away, as I have noticed quite frequently in the Commodore newsgroups on the Internet's usenet of people asking for copies of long dead games and applications. Several times I have had to send a curt E-mail reply to several Commodore users in Europe seeking pirated copies of GEOS version 2.0 and warn them that they were in violation of American and international copyright laws in distributing software in this manner.

Because of the selfish actions of pirates like these, all of the Commodore 64/128 users have suffered. I also should point out that software piracy is not just limited to the Commodore computers either. Already, we are starting to see the impact that software piracy is having on Amiga computers, as well as IBM types and MacIntoshes. It is becoming such a problem that the software industry had to have Congress amend the copyright laws to protect themselves and the software producers have put pressure on federal agencies to enforce those laws protecting intellectual properties.

As one who writes professionally, there is nothing more hideous than someone who would steal my work and not compensate me for it. If I were a programmer, I would feel the same way, particularly after putting in hours creating a program, utility or game, hoping to make a profit, only to find somebody else stealing that work and distributing to others because of a twisted sense of morality. Particularly those individuals who feel that he or she has a right to my work without compensating me for it. In my book that's outright theft and it's wrong.

So, if you still are pirating copyrighted software, do us C-64/128 users all a favor and knock it off. After all, we still want to enjoy our favorite eight bit computers well into the 21st Century, don't we?

Harold Stevens, Jr. is an avid Commodore 64 and GEOS user and is the Friday night Commodore 64 Roundtable Conference (RTC) host on Genie. His internet address is hstevens@freenet.columbus.oh.us

USED SOFTWARE

We buy, sell, and trade used original software. Lowest prices for C64, C128, Amiga, and IBM. Mention this ad and your computer type for a free list. Call our BBS for a complete list within minutes.

BBS:
8N1 1200-28.8k
616-429-7211

Ask for a list via Internet:
CENTSIBLE@DELPHI.COM

Call or write:

CENTSIBLE SOFTWARE

P.O. Box 930
St. Joseph, MI 49085
Phone: 616-428-9096

Sample: $2.95
6 issue sub:
$14.95

the Gatekeeper
610 First St.
Liverpool
NY 13088

Write programs in C, build a lightpen, know all about the latest products like 4-Scan, Novaterm 9.6 and 64NET, and learn how to FLI! Advanced 64/128 users want more. We have it.

The final page in commodore computing.

COMMODORE COUNTRY

Everything for Commodore Computers
Sell • Trade • Repair • Buy
1420 County Rd. 914
Burleson, TX 76028
817-295-7658
817-447-6974 - Voice/FAX line

ALL STORE ITEMS HAVE A 90 DAY WARRANTY. IF, FOR SOME REASON YOU ARE DISSATISFIED IN YOUR PRODUCT, YOU MAY RETURN IT FOR REPLACEMENT OR STORE CREDIT ONLY. SORRY, NO REFUNDS.

C64/1541 Repair - $40
SPECIAL
C64, 1541, J-S+Sft. - $99

We carry a full line of hardware, software & magazines, both new & used, including European items. Our flat-rate repairs include both parts & labor. Call for details. Trade in your unwanted items.

Catalog - $2.95
ALL MAJOR CREDIT CARDS ACCEPTED.

We carry a full line of Amiga Products.

CREATIVE PIXELS PRESENTS
New for the C-64 / 128

The Ghost of Genghis Khan

- ***Movie Style Storytelling**
- ***Super Graphics**
- ***Arcade Action**
- ***One or Two Players**

Also Available:
THE WINGS OF CIRCE
ESCAPE ROUTE
ISLAND OF THE DRAGONS
PREHISTORIC MAN

Send check or money order for $19.95 to:
CREATIVE PIXELS LTD.
P.O. Box 592
Library, PA 15129

UPGRADE YOUR COMMODORE!!!

Refurbished Hardware

Monitors		Drives		Other	
1701	$149.95	1541/C	$89.95	C64 -	$94.95
1702	$149.95	41 w/Dips	$104.95	64C -	$114.95
1802	$174.95	1541-II	$119.95	128 -	$174.95
1802D	$199.95	1571	$159.95	128D	$324.95
1902	$224.95	1581	$174.95	SX-64	$349.95
1902A	$249.95	MSD-2	$149.95	1660	$24.95
1084	$294.95	1001SFD	$129.95	1670	$39.95
1084S	$324.95	B.I. Buscard-II			$59.95
Mono's	$44.95+	1530 Datasette			$34.95

Miscellaneous

Books$10-$15	Printer Interfaces	$49.95+
Prog's $5-$20	Epyx Fastload Cartridge	$34.95
Repairs $ASK	Super Snapshot v4	$49.95
Manuals $7	Super Grafix Jr.	$39.95

New Hardware

New APROTEK Accessories

C24-2400 Baud (64/128)	$74.95
User Switch	$30.95
Convert-A-Com	$29.95

New CMD/LMS Accessories

JiffyDOS C64/SX64 'System'	$69.95
JiffyDOS 128/128D 'System'	$79.95
128 Kernal $59.95 64 Kernal	$49.95
Additional JiffyDOS Drive ROM	$34.95
RAMLink Bs. $209.95 c/w 0 MB	$274.95
c/w 1 MB $319.95 c/w 4MB	$479.95
Real Time Clock (Optional) Add	$24.95
FD-2000	$249.95
NEW Super Snapshot v5.22	$84.95
1750 Superclone REU 512K	$169.95

Ask For Anything! We May Have It!

J.P. PBM Products By Mail
Box# 60515, N. Sheridan Mall P.O.
Downsview, Ont. Canada M3L 1B0
Tax-Canada + 7%GST, Ontario +8% PST

Send CDN Funds/15% USA Exchange
15 day Warranty On Refurbished Hdwr.
Allow 4-6 weeks for delivery
1996 Catalogue Disk (64 Format) — $2

Shipping-($0-$25=$4, $26-$99=10%, $100-$199=8%, $200-$499=7.5%, %500=-6%, USA=15%)

FREE CATALOG

With Tons of Commodore Stuff!

Write to: FREE CATALOG
Parsec, Inc.
P.O. Box 111- CW
Salem, MA 01970-0111
USA

CW & RUN BACK ISSUES

For years, *RUN* Magazine provided Commodore Users with a great source of information, and now CMD has given you *Commodore World*. Don't let this valuable information slip away—fill in the voids in your library now!

RUN Magazine Back Issues

Any 3 issues for $12.00, any 6 for $18.00, or any 12 for only $24.00

RMJAN89	January 1989	**RMJUN90**	June/July 1990
RMFEB89	February 1989	**RMDEC90**	December 1990
RMMAY89	May 1989	**RMJAN91**	Jan/Feb 1991
RMJUN89	June 1989	**RMJUN91**	JuneJuly 1991
RMJUL89	July 1989	**RMJAN92**	Jan/Feb 1992
RMAUG89	August 1989	**RMMAR92**	March/April 1992
RMSEP89	September 1989	**RMMAY92**	May/June 1992
RMOCT89	October 1989	**RMJUL92**	July/August 1992
RMSP89	Special Issue 1989	**RMSEP92**	Sep/October 1992
RMJAN90	January 1990	**RMNOV92**	Nov/December 1992
RMMAY90	May 1990		

Shipping: 3-6 issues - U.S. $3.00, Canada $5.00, Foreign $15.00; 12 issues U.S. $5.00; Canada $7.00; Foreign $20.00.

Commodore World Back Issues

$4.95 each, or any 3 for only $12.00

CW1	Commodore World Issue 1	**CW9**	Commodore World Issue 9
CW2	Commodore World Issue 2	**CW10**	Commodore World Issue 10
CW3	Commodore World Issue 3	**CW11**	Commodore World Issue 11
CW4	Commodore World Issue 4	**CW12**	Commodore World Issue 12
CW5	Commodore World Issue 5	**CW13**	Commodore World Issue 13
CW6	Commodore World Issue 6	**CW14**	Commodore World Issue 14
CW7	Commodore World Issue 7	**CW15**	Commodore World Issue 15

Shipping: U.S. and Canada $2.00 for first issue, plus $1.00 per additional issues.; Foreign $5.00 per issue.

TO ORDER CALL 1-800-638-3263

* CLASSIFIED ADS *

C64/128 PUBLIC DOMAIN. REQUEST FREE CATALOG OR SEND $2 FOR A DEMO & CATALOG. CALOKE IND., P.O. BOX 18477, RAYTOWN, MO. 64133. VISA-M/CARD ACCEPTED.

C=Hacking in Print! $6.00 Not simply a printout! Jim Brain,10710 Bruhn Ave, Bennington, NE 68007. Printout of COMP.SYS.CBM FAQ: $12.00. Prices include shipping.

Geos Publication. The exclusive U.S. monthly GEOS publication. $14.00/year, $25.00/2 year. Write: 713 E. Main St., Independence, KS 67301-3726. Phone: 316-331-2487.

The Ultimate System III (Lottery Prgram), The Ultimate Grade Book (For teachers) $19.95 each. To Order: CALL (202-316-1803/Tony) Write: A. Tolson, P.O. Box 62326, Wash., DC 20029-2326.

CMD 4MB RAMLink plus many assorted games. Excellent Cond. $130 for all. 815-259-2816 or write: 3636 Grandview Pl., Thomson IL 61285

For Sale: (2) Vic 20 computers w/ PS, (1) 1541 Drive w/ Cables, (1) 1530 Datasette. Best Offer. George York: (501) 481-5682 after 4 pm (CT)

For Sale: C-128/1571/1902A Monitor, Copal Printer, 300 modem, C64/1541, 30+ assorted software (educ./games/office) - extras. All excellent condition. $275 O.B.O. 804-744-0165

HELP! I have "Titanic-The Recovery Mission" and I'm Stuck! Can anyone help me, Please? Robert Wallace, 131 White Horse Park, Absecon, NJ 08201-2402

D-64, DD $79.95, C-128, 1571 $119.95, A500 $125, Cables, Monitors, Software! Send SASE for complete list. Dept C315, POB 10121, Eugene, OR 97440.

C-128D, 2nd 1571, 1084S, SP-1000VC. New Unopened: GEOS 128 2.0, CadPak 128, WordPro 128. Many Extras. Works Great. $500.00 Delivered. (502) 828-8328.

FREE COMPUTER MAGAZINES! You pay the shipping. For list send SASE to: James Lambert, 1121 Williams Ave., Hastings, NE 68901-3820.

Don't wait until it's too late...

RENEW EARLY!

Is your Commodore World subscription close to running out? Here's an easy way to check: look at the mailing label on the front of your copy. You'll find your subscription number and the expiration issue number. For example:

```
James Smith  12345EXP17
123 Home Street
Grand Rapids, MI  49502-0123
```

Jim's subscription will run out with Issue 17 as indicated by the EXP17 in his subscription code. Jim would be wise to re-subscribe early to avoid missing a single issue of Commodore World!

ADVERTISERS INDEX

COMMODORE WORLD Classified Advertising

Commodore World subscribers may place non-commercial classified advertising in *Commodore World* at a cost of $10.00 per issue. Your advertisement may contain up to 150 characters (including spaces). Send your advertisement with payment to: CW Classified Advertising, c/o Creative Micro Designs, Inc., P.O. Box 646, East Longmeadow MA 01028-0646.

CMD Service Center

AFFORDABLE • FAST • DEPENDABLE

Call Today 1-800-638-3263

We repair the following equipment: Commodore C-64, 64C, SX-64, C-128 and C128-D computers; 1541, 1541C, 1571 and 1581 Disk Drives plus CMD Devices. JiffyDOS Installations a specialty. All repairs warranted for 30 days. Minimum charge $35.00 plus parts and return shipping. You must contact CMD for authorization before sending any equipment. **Creative Micro Designs, Inc. P.O. Box 646 E. Longmeadow, MA 01028**

Enter the

COMMODORE WORLD

Sweepstakes

The Commodore C-65, also known as the C-64DX, was Commodore Engineering's greatest 8-bit achievement, but the project was cancelled before the machine was fully debugged. However, Commodore did create about 200 'Alpha' units, early versions of the computer which were to be sent to developers for testing. If you're our lucky Grand Prize winner, you can own one of these rare pieces of Commodore history—for the price of subscribing!

For more than two years, Commodore World has empowered C-64/128 users by providing the latest news and information on the world of Commodore computing. Through our staff of leading experts like Jim Butterfield, Maurice Randall and Gaelyne Moranec, we keep you in touch with new technologies and further explore those features which have made the Commodore great! So — subscribe or renew today for your chance to win big in Commodore World's Sweepstakes '96.

Call 1-800-638-3263

and subscription or renew now!

Contest valid on new subscriptions or renewals. Must be received by September 30, 1996 to be eligible for the sweepstakes. Winners announced in CW Issue 18. Contact CMD for complete rules.

Grand Prize:

Your choice of: a Commodore C-65 computer or 1 MB Super CPU

(2) Second Place Prizes:

Super CPU Accelerator

(12) Third Place Prizes:

$10. CMD Gift Certificates

Please begin my subscription to Commodore World as soon as possible, and enter me in the sweepstakes!

Subscriber Information

Name: ____________________

Address: ____________________

City: __________ State/Prov. _____ ZIP/PC: ________

Country: __________ Phone: (   ) __________

One Year Subscription (Eight Issues)

United States..............$29.95	Canada & Mexico......$35.95
Europe (EC Only)......$45.95	Other Foreign................$57.95

Back Issues:

Single issues $4.95 each plus $2.00 shipping. Three or more back issues $4.00 each plus $1.00 each for shipping and handling.

Payment Information

❑ Bill me later (one year subscriber only)

❑ I've enclosed a check or money order in U.S. Funds

❑ Bill subscription to my credit card: (Check one)

❑ Visa ❑ MasterCard ❑ AMEX ❑ Discover

Card Number: ____________________ Exp. Date: ______

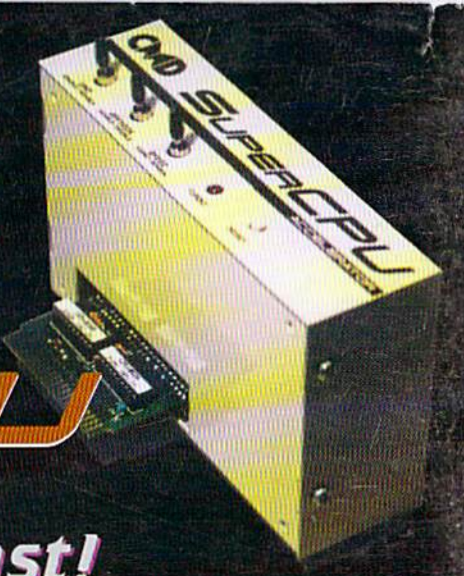
Signature: ____________________

Mail payments to:

Creative Micro Designs, Inc.
CW Sweepstakes '96
P.O. Box 646
East Longmeadow, MA 01028

20 MHz on a C-64? Sure...

WHEN PIGS FLY!



SuperCPU

Ultra Fast!

Compatible!

Easy to Use!

Starting at

$199.00

plus $7.50 S&H

To order call:

1-800-638-3263

(Mon.-Fri. 9 a.m. — 5 p.m. EST)

CMD

64 Model Available Now!
128 Model Coming Soon!