

The Transactor

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

Issue 05
\$2.95

Special Reference Issue Everything For Plugging Into Your Commodore Equipment

And All In One, Handy Place! Contains:

- BASIC and Machine Language Summaries
- Screen, Sprite, and Character Design Grids
- New Butterfield SuperCharts
- Memory Maps For All Machines
- Disk Specs For All Drive Types
- Port Pinouts For All Machines
- Colour and Sound Tables
- Book List and Glossary
- ROM Entry Points
- Plus LOADs More!



Richvale Telecommunications

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

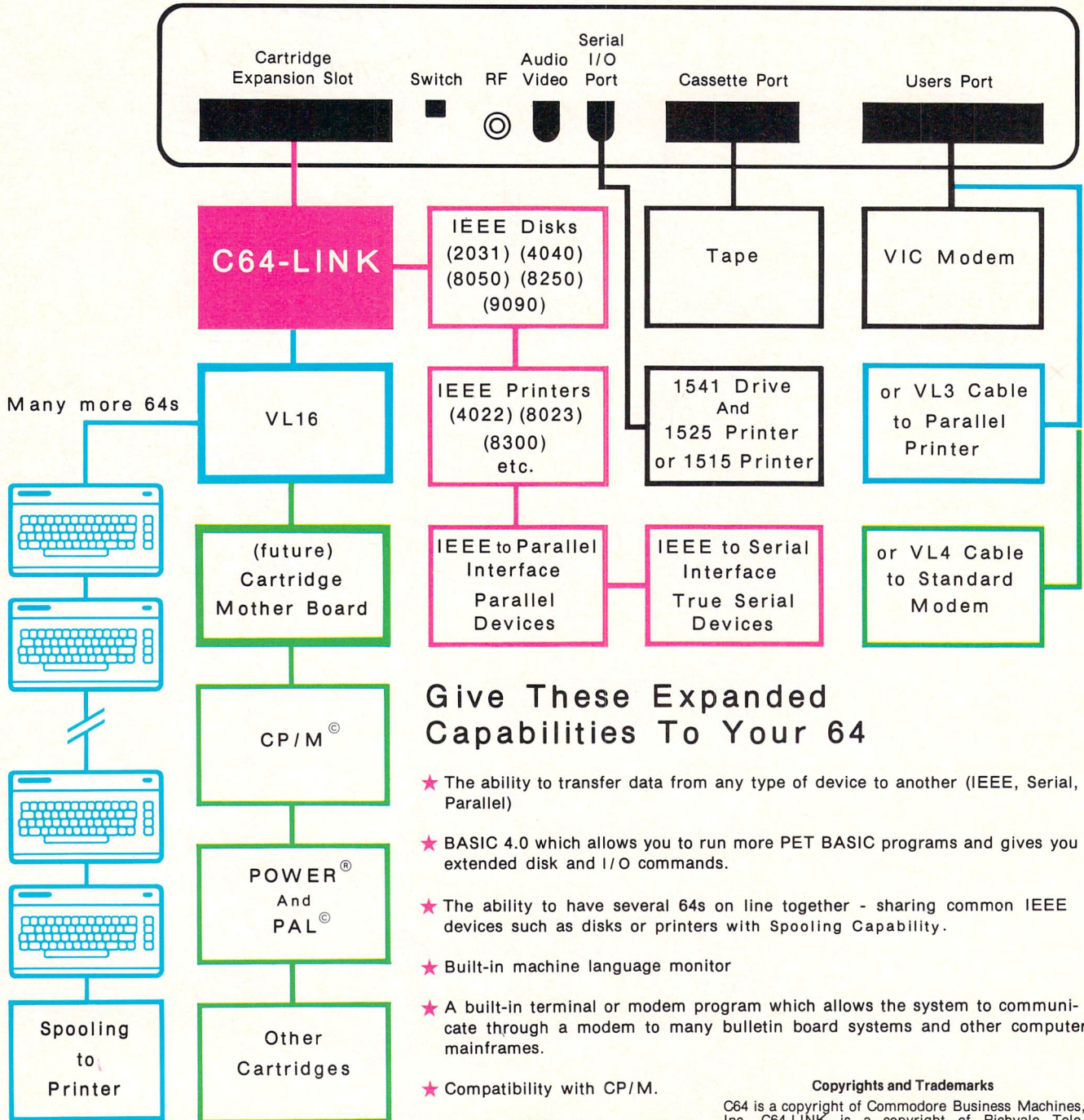
\$185 Canadian

C64-LINK[®] The Smart 64

Also available
 for VIC 20

RTC

RTC



Give These Expanded Capabilities To Your 64

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

Copyrights and Trademarks

C64 is a copyright of Commodore Business Machines, Inc. C64-LINK is a copyright of Richvale Telecommunications. CP/M is a registered trademark of Digital Research. POWER is a trademark of Professional Software. PAL is a copyright of Brad Templeton.

Contact your local Commodore dealer or RTC.



nüFEKOP

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

1-800-525-2529



Software for the VIC 20™
and Commodore 64™

The Transactor

Editorial	5
News BRK	6
The MANAGER Column	10
CompuKinks	15
Advertising Section	70
Advertising Index	88

Reference Section

BASIC

Commands and Statements	16
Arithmetic Operators	18
Special Symbols	18
Reserved Variables	18
Hierarchy of Operations	18
BASIC 4.0 Disk Commands	19
Additional B Series Commands	19
String Functions	20
Arithmetic Functions	20
Mathematical Functions	20
Status Variables – ST, DS & DS\$	21
Secondary Address Table	21
Printer Control Characters	21
8032 Screen Control Characters	21
Error Messages	22

Screen Visual

Sprite Design	24
Programmable Character Design	25
PET/CBM 40 Column Screen Map	25
VIC 20 Screen and Colour Table Maps	26
Commodore 64 Screen and Colour Table Maps	28
VIC 20 Screen Memory Addresses	29
VIC 20 Character Base Addresses	29
Commodore 64 Screen Memory Addresses	29
Commodore 64 VIC II Chip Addresses	29
Commodore 64 Character Base Addresses	29
Character ROM Partitioning	29
CBM (8032, 8096, SuperPET) Screen Map	30
B Series 80 Column Screen Map	31
VIC 20 Screen and Border Colours	32
6845 Video Chip Registers	32
VIC 20/Commodore 64 SuperChart	34
BASIC 2.0/4.0 SuperChart	36

Volume 4, Issue 05

The Reference Issue

Sound	
Commodore 64 SID Note Values	33
Commodore 64 ADSR Envelope Values	33
VIC 20 Note Values	33
Memory Maps	
BASIC 2.0/4.0 Zero Page Contents at Power-Up	38
VIC 20/Commodore Zero Page Contents at Power-Up	40
BASIC 2.0/4.0 RAM, ROM, and I/O	42
VIC 20 RAM, ROM, and I/O	44
Commodore 64 RAM, ROM, and I/O	46
B Series RAM, ROM, and I/O	48
VIC 20 Super Expander Map	50
65XX Machine Language	
Hexadecimal Conversion Table and Bit Values	32
Wallet Size Op-Codes Chart	32
Instruction Set Summary	52
Instruction Set – Alphabetical Sequence	54
Addressing Modes	54
BASIC 2.0/4.0 Kernal Routines	55
VIC 20/Commodore 64 Kernal Routines	55
User Callable ROM Routines	56
Keyword Tokens and Entry Points	57
Input/Output Ports	
VIC 20/Commodore 64 RS-232	58
IEEE-488 Bus Signal Descriptions	59
IEEE-488 Port	59
PET/CBM User Port	59
Cassette Port	59
VIC 20 Expansion Port	60
VIC 20 User Port	60
VIC 20 Audio/Video Port	60
VIC 20/Commodore 64 Serial Port	60
VIC 20/Commodore 64 Joystick Ports	61
Commodore 64 Expansion Port	61
Commodore 64 User Port	61
Disk Drives	
Disk Specs	62
Utility Command Set	62
User Command Jump Table	62
Block Allocation Map Formats	63
Directory Header Formats	63
Directory Block Formats	63
Commodore Related Book List	64
Glossary of Terms	66

The Transactor

The Tech/News Journal For Commodore Computers

Managing Editor

Karl J. H. Hildon

Editor

Richard Evers

Advertising Manager

Kelly M. George
416 826 1662

Art Director

John Mostacci

Contributing Writers

Greg Beaumont
Don Bell
Dave Berezowski
Jim Butterfield
Gord Campbell
Chuan Chee
Mike Donegan
Alex Gaul
Donna Green
Fred Hambrecht
Paul Higginbottom
Dave Hook
Eike Kaiser
Peter Lear
Bill MacLean
Darren Spruyt
John Stoveken
Brad Templeton
Don White

Production

Attic Typesetting Ltd.

Printing

Printed in Canada by
MacLean Hunter Printing

Program Listings In The Transactor

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

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

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

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

Cursor Characters For PET / CBM / VIC / 64

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

Colour Characters For VIC / 64

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

Function Keys For VIC / 64

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

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

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

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

Canadian Distributor:
Access Computer Services
620B Magnetic Drive
Downsview, Ontario
M3J 2C4
(416) 736 4402

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

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

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

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

Editorial contributions are always welcome and will appear in the issue immediately following receipt. Remuneration is \$20 per printed page. Preferred media is 2031, 4040, 8050, or 8250 diskettes with WordPro, WordCraft, Superscript, or SEQ text files. Program listings over 25 lines should be provided on disk or tape. Manuscripts should be typewritten, double spaced, with special characters or formats clearly marked. Photos of authors or equipment, and illustrations will be included with articles depending on quality. Diskettes, tapes and/or photos will be returned on request.

All material accepted becomes the property of The Transactor. All material is copyright by Transactor Publications Inc. Reproduction in any form without permission is in violation of applicable laws. Please re-confirm any permissions granted prior to this notice. Solicited material is accepted on an all rights basis only. Write to the subscriptions address above for a writers package.

The opinions expressed in contributed articles are not necessarily those of The Transactor. Although accuracy is a major objective, The Transactor cannot assume liability for errors in articles or programs.

From The Editor's Desk

Infolution

That's right. Infolution - the evolution of information. Or is that revolution of information. In 1977 (here I go again) when Commodore first released the PET, there was virtually no information available. Then as more questions were asked, more material was generated. Soon there was enough to get by, and after three years plenty for most purposes. But it was scattered and I found myself carrying 20 pounds of stuff around to have maybe 20 pages of it with me. So in 1980, The Transactor released the first Reference Issue, Volume 2, Issue 7.

History repeats itself. When I noticed my briefcase was splitting at the seams, I figured it was time. It would have to be complete, accurate, tidy, and most of all, handy. But in the past 6 years, a lot has been covered, and re-uniting it all into a compact unit would prove to be some task.

Somewhere in the neighbourhood of 2000 hours was spent on this orb. Usually an issue takes about 1 1/2 of the typesetters' 8" disks. This time 6 of them were completely filled. But this one makes the last one typeset by yours truly. Hopefully this will shave some time off our cycle.

You will notice a certain amount of duplicated material as you flip through for the first time. Quite intentional... and the reasons will become more apparent as you use it more. For example, you'll find machine language op-code values presented in at least three spots. Each has its own purpose in life. Butterfield's SuperCharts show the relationship between values as they're interpreted by ASCII, BASIC, the screen, and the microprocessor. I find this chart is most useful when trying to decipher someone else's work (and also anything of my own that's over three months old).

When I'm writing new code, I usually have the 6502 Summary at hand. Alphabetical order makes the instruction I need easier to find, plus the applicable addressing modes and affected status register bits are the kind of information one must have under such circumstances.

The op-code hex values appear again in a rather handy wallet-size chart drafted some years ago by Jim Butterfield. However, if you look between the lines on this one, you'll see a pattern. It was this chart that first introduced me to the structure of the 6502's internal architecture. Notice how 90% of the instructions have been given very deliberate values, not just arbitrary.

Further on the above, the Reference Issue has been assembled with a "same page" philosophy. I admit the characters

are a bit small in places, but I find the inconvenience of flipping pages is worth the tradeoff.

Once again I salute Jim Butterfield, the man responsible for all those memory maps and other stuff. The hours of trouble he has saved us all could easily be measured in lifetimes. By the way, we still have copies leftover of Issue 02 featuring a centerfold of the man himself... and in fine form, I might add.

Last, I should point out that this issue was designed to consolidate the most frequently used information. It is by no means a total guide. For more details on the material presented here, you should consider the 'VIC 20 Programmers Reference Guide', the 'Commodore 64 Programmers Reference Guide' (both from Commodore), or 'Programming The PET/CBM' by Raeto West (published by Compute). All 4 would make a truly complete anthology.

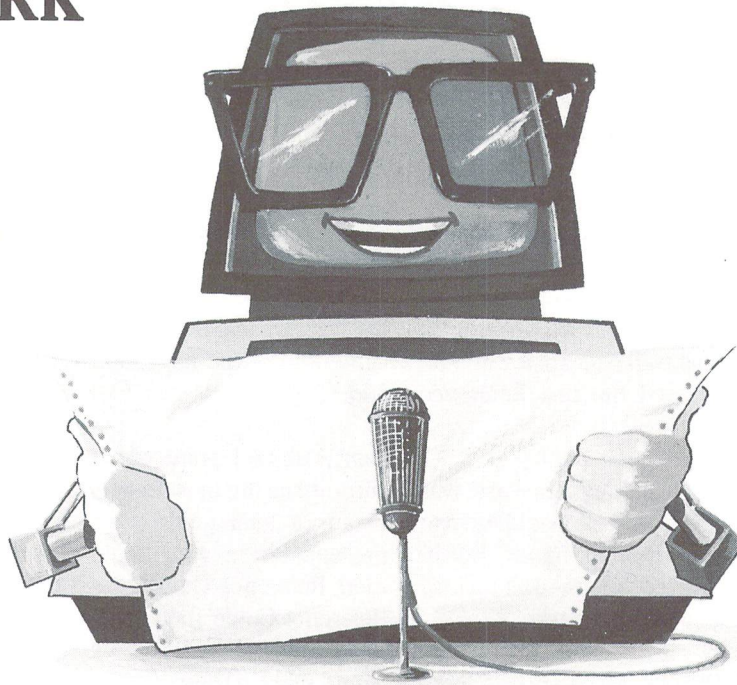
And in closing... this issue marks our debut on the newsstand in Canada - we printed 16,000 to meet orders, more than in all of 1979 and '80. And now that typesetting detail is being farmed out, Richard Evers, new full-time Editor, and myself can concentrate on cranking out files instead of film. More writers, articles, circulation, and faster production is now on top of the priorities list. We're paying more now too, call us.

No longer Editor, no longer typesetter, I remain,



Karl J.H. Hildon
Managing Editor, The Transactor

News BRK



The Transactor Adds New Employees And New Features

Richard Evers is now full-time Editor of The Transactor and Karl Hildon moves to the Managing Editors' spot. Both Richard and Karl will continue working on article production. Adding to the traditionally advanced technical content of The Transactor will be more material at the beginner and intermediate levels. This does not mean that advanced level material will be reduced, simply that beginner and intermediate will be increased. This will help broaden the reader base of The Transactor and ultimately lead to increases at ALL levels.

Renewal Time

For many subscribers, this will be the last issue posted against their subscription since The Transactor became independent in November 1982. This notice applies mostly to those that sent less than \$15 towards a subscription. If you are one of them, your mailing label will show an asterisk (*) at the top right corner.

If you subscribed starting with Issue 01 and sent the right amount, your last issue will be the next one, Issue 06. However, the postage-free subscription card will NOT be run next issue, so take advantage now. Please indicate 'Renewal' on the card.

No More 04's

Unfortunately, we have run out of Volume 4, Issue 04. Yes, for the first time, we plain ran out! If there is enough demand, we'll print more, but those ordering new subscriptions should consider Issue 05 as their first copy.

Prices in US. Dollars

Lately we've run a couple of ads that show prices with no indication of their currency, ie. Canadian or US. dollars. If the ad has been placed by a firm with a US. address, it should be assumed that the prices are in US. dollars. In future we'll attempt to clarify any ads that don't specify.

COMAL For The 64 Now Public Domain

As of September 30/1983, COMAL for the Commodore 64 has been free public domain software. COMAL was originally written for the PET/CBM. This new rendition has several improvements including commands for controlling colour, sound, and turtle graphics. Best of all, it's free. Contact your club librarian about obtaining copies.

'Home Accountant' and 'Tax Advantage' for C64

Continental Software of Los Angeles, California, in conjunction with their recently appointed marketing firm, South Western Publishing Co., has announced the release of two of their current best selling programs, 'The Home Accountant' and 'The Tax Advantage', for the educational environment. Rewritten specifically for the junior high schools, high schools, universities and adult education classes, the packages all come complete with diskettes, users manuals, teachers manual and test forms. For more information contact:

Hank Scheinberg/Denny Mosier
Continental Software
11223 South Hindry Avenue
Los Angeles, California
90045 (213) 410-3977

Las Vegas Style Gambling On The C64

Advanced Microware has announced the release of their latest effort for the C64, CASINO PAC. It includes four games - Blackjack, Poker, Keno and Slot Machine, and each is said to be accurate simulations of the video game machines currently in use in Las Vega, Reno and Atlantic City. Available on tape or disk for \$39.00 US.

Also available is 64TOUR, a demonstration program of the capabilities of the C64. These include High Resolution Graphics, Music and Sound Effects. Priced at \$12.00 US. For more information please contact:

Advanced Microware
P.O. Box 6143
Santa Ana, CA.
92786 (714) 554-6470

Save The World From Total Destruction With Your Vic 20

THORN EMI has announced the release of their newest and most challenging game yet, 'Computer War', based on the movie 'War Games'. Though difficult to master, it has been made easier to learn through all stages. Control is accessed either through the keyboard, or with a joystick. For more information please contact:

Nina Budman or Pat McNamara
Budman Math Inc.
505 Eglinton Avenue West, Suite 303
Toronto, Ontario
M5N 1B1 (416) 484-7773

SOLARCON announces 'Passolar', Design Analysis For Passive Solar Systems

Based on the 'Passive Solar Design Handbook', Volume Three from the Los Alamos National Laboratory, Passolar has been hailed as an easy to learn, easy to operate Architects, Consultants and Contractors tool. With input data provided or generated by the client, your desired results are easily attained with the use of this program. Instructions are shown on the screen while the program is in operation, and a very comprehensive users manual is also provided. Available for the Commodore series of business computers, the prices are as follows. \$355.00 US with North American Weather Data, and \$525.00 US with World Wide Weather Data. For more information please contact:

Norlene H. Martin
SOLARCON, Inc.
607 Church
Ann Arbor, Michigan
48104 (313) 769-6588

Spike-Spikers™, Voltage Suppressors & Noise Filters

Kalglo Electronics Co., Inc. has announced their release of a line of eleven different models of transient voltage suppressors and conducted power line noise filtering devices. Designed to protect, control and organize sensitive high-tech equipment, these units are available at prices ranging from \$34.95 to \$94.95 US. For further information please contact:

Kalglo Electronics Co., Inc.
Dept. SS
6584 Ruch Road., - E. Allen Twp.
Bethlehem, PA 18017

C.Itoh F10 Printer Cut Sheet Feeder Available For Under \$900.00

Canadian General Electric has announced the arrival of its fully mechanical cut-sheet feeder for its C.Itoh F10 Daisy-wheel Printer. The Model A-100 requires no tools for installation, no power to operate and has no switches or controls at all. Weighing in at ten pounds, this completely metal unit will stack in correct sequence each piece of paper fed to it from its 200 sheet storage bin. Subscripts and superscripts are supported, with a price in total being \$850.00 Cdn. For more information:

Mr. Art S. Best
Canadian General Electric - Data Systems
(416) 675-7500

The Grade Manager

'The Grade Manager', for the Vic-20 and the Commodore 64 computers, will alphabetically sort student lists, keep track of assignments, weights, grades, and calculate averages for an entire term. It will report, to TV or printer, assignment summaries, student grades and averages, and incomplete assignments. Available on both cassette and diskette, the disk version is menu driven. For use with the Vic-20, 8K memory expansion is required. Prices are \$29.95 US for tape and \$34.95 US for disk. For more info, contact

Smokey Mountain Software
54 West Main Street
Brevard, NC
28712 (704) 883-2595

Two New Sports Strategy Games For The VIC 20

Parr Programming, of Gary, Indiana, has produced the first two in a series of Sports Adversary Games for the Vic 20. The first is the Baseball Adversary. With you in the managerial position against your computer, this game allows for additional players on the bench for pinch hitting and relief pitching and is written to not allow your adversary, the computer, any unfair advantage. Control of the game is through the keyboard. Available on cassette and requiring no memory expansion, the price is \$10.95 US.

The Football Adversary is the final in this series of two. You are put in the role of head coach, with features being a selection of 31 plays and a display of halftime/game statistics. Available on cassette, and also using keyboard control, the price is \$14.95 US.

Parr Programming
2664 Tyler Street
Gary, Indiana
46407 (219) 885-0611

Kobetek Systems Limited announces new STATS software package.

Designed by Mr. Patrick Royston of London, England, Mr. Royston currently holds a masters degree in both statistics and information science. Written for the 4000, 8000/9000 and C64 series of computers, this menu driven program gives the user a completely advanced statistical program for use in any STATS application. With a price of \$350.00 to \$500.00 Cdn., further interest may be warranted for those who require accurate results in a shorter period of time. For

more information contact:

Kobetek Systems Limited
1113 Commercial Street
New Minas, Nova Scotia
B4N 3E6 (902) 678-9800

SWL RTTY/MORSE - Attention Hams and Short-wave Listeners

Let your Commodore 64 or VIC-20 help you 'listen in' on the world of Radioteletype and Morse Code. Converts your VIC-20 and C-64 computer into a low-cost RTTY and morse decoding and display terminal. Allows you to receive 45 to 100 baud BAUDOT RTTY and 5 to 30 wpm morse code. Requires TTL compatible terminal unit such as the MFJ-1224/5 or the HRA Electronics TU-II. Package includes software on cassette, special user port connector, and complete instructions. Price \$19.95 US for cassette or \$22.95 US for disk plus \$2.00 for shipping and handling.

RAK Electronics
P.O. Box 1585
Orange Park, Florida
32067-1585 (904) 264-6777

Modem Protection From Kleen Line

Suppress damaging telephone and power line spikes caused by lightning, spherics or phone office switch gear with the Kleen Line Model PDS-11/SUP. Available for 4 pin telephone modular connectors (RJ-11) and the wider 8 pin connectors (RJ-45), the PDS-11/SUP is priced at \$81.95 US. For more information please contact:

Frank Stifter
Electronic Specialists Inc.
171 South Main Street
P.O. Box 389
Natick, Massachusetts
01760 (617) 655-1532

Accounts Receivable Program For The C64

Written by Microspec Inc. of Plano, Texas, this program will support up to 150 customers per diskette on a balance forward system that will accommodate 18 transactions per customer for each posting period. Complete aged reports are generated and automatically posted to Microspecs General Ledger program. The price for this program is \$99.95 US. with more information available from:

Laura Robinson
Communications Plus
301 S. Sherman
Suite 117
Richardson, Texas
75081 (214) 783-8543

Allow Your Vic or C64 To Balance Your Chequebook

Microspec of Plano Texas, in its never ending quest for further C64 and Vic 20 programs, has released the 'Checkbook Manager'. Available for both the C64 and Vic 20, each appear to offer a solution to the problem of keeping your chequebook in order. Able to track all outstanding cheques, reconcile monthly financial statement, and track expenses on a quarterly and year-to-data basis, the limit only appears to be the capacity of the version purchased. The Vic 20 version offers ten less categories than that of the C64. The price is \$69.95 US for the C64 version, and \$49.95 US for the Vic 20. For further information please contact Communications Plus (above).

'Power 64' Available For The Commodore 64

In the past we have all come to know and love the features of Power for our Commodore computers. But with the advent of the Commodore 64, a loss has been felt. After being spoiled by Power, BASIC programming has suddenly become a chore. No longer must this be, for Brad Templeton has re-written his utility for the C64.

In case you have never heard of Power, the following are some of the features it has to offer.

- automatic line numbering and re-numbering
- complete tracing functions
- single stepping through programs
- definition of keys as BASIC keywords
- de-bugging ease with a why command
- addition of auto repeat function to cursor keys
- ability to merge BASIC programs
- listing and scrolling up and down of BASIC programs in memory
- hexadecimal and decimal conversions
- and much, much more

The documentation included, written by Jim Butterfield, is very helpful in all aspects and leaves nothing to the imagination. For a total price of \$99.95 Cdn., the benefits shown will be worth the cost involved. For more contact:

Pro-Line Software Ltd.
755 The Queensway East, Unit 8
Mississauga, Ontario
L4Y 4C5 (416) 273-6350

The MANAGER Column

Don Bell & Alex Gaul
Milton, Ontario

In this issue, the Manager column is devoted to the first part of a hands-on application for the '64 MANAGER - a Christmas card and gift list. The second part of this article will appear in the next issue. Subsequent issues will provide more easy to follow hands-on applications for home and business as well as tips for advanced users including "Re-Field" by Eric Armson, a routine you add to Create/Revise that re-orders field entry. - Ed.

A '64 MANAGER Gift List

HUMBUG! It's that time of year again. I staggered up the stairs from the basement dragging the box of old Christmas junk accumulated over years of neglect. Then after a stiff drink, I screwed up enough courage to open it. YUK! Krinkled tinsel, tangled Christmas tree lights, broken decorations, horrible-looking holly, musty mistletoe (smelling of old kisses), petrified candy canes and at the bottom, boxes of last year's Christmas cards. Cheer up, the worst is yet to come!

Then, like a memory of a NEW Year's Day hangover, it suddenly hit me! How am I going to cope with the Christmas cards, gifts, and party again this year? There's a number of people I'd like to 'edit' from my list but I can never remember whether they sent me a card or a gift last year. (I'd prefer not to expose myself as an incompetent who can't even master a Christmas card list.) Usually what happens is I send them a card one year, they send me one the next... and so on. It's as though each of us wants to send the last missile... I mean card. Then what to do about gifts - especially those useless token gifts to appease distant relatives. I've got enough socks & ties to dress a herd of caribou and enough after-shave lotion to bathe in. I'd trade it all for one good Darth Vader video game. It gets down to a matter of how much to spend - an equation - whatever they spend I have to spend. Boring! Then who to invite to the party? How to avoid inviting people who mix like sharks and jellyfish. How will I get through it all again without losing my mind?.

HARK! This year I decided there had to be a better way. I want to get rid of the worry, the planning, the strain on my old memory bank and get back some real Christmas cheer,

brouhaha, whatever it was that made old Scrooge want to click his heels on Christmas Day. Taking my cue from a business associate of mine who is ecstatic over his new computer system, I decided to try my hand at solving my problem using THE '64 MANAGER.

Very slowly and carefully I will show you the steps I took creating my Christmas List file. You will be able to easily keep track of cards, gifts, parties, money spent, and other Christmas information. You will be able to generate numerous useful reports that would be difficult or impossible to create without it. All of this is directed towards helping you better organize yourself, so that you will have more time to experience the real joys of Christmas.

LOADing The Program

Turn on your 64, monitor and disk drive.

Insert your Manager disk in the drive

Type: load "64.manager", 8 press 'return'.

When the cursor reappears, type in 'run' and press 'return'.

After a few minutes, a menu comes up on the screen and you are ready to start.

Formatting A Diskette

The first thing you must do is "format" a blank diskette, ie. prepare it so it can receive data. Press 'f' to format a disk. The computer will load the format program from the disk. When the computer has finished loading, remove THE MANAGER diskette and insert your blank data diskette into the drive. Follow the screen instructions and you will have a formatted disk.

Getting Started In The CREATE/REVISE Option

When the main menu appears, press 'c' for create/revise. This option lets you draw a form on the screen and revise it later if you choose. The first screen prompt is 'filename?'. I called my file 'xmas list'. Enter any filename you want (preferably one that reminds you of the file) and press 'return'. The next screen prompt is 'create using an existing screen (y/n)?'. Enter 'n' as you do not have an existing screen.

Selecting Screen Colours

Since this is a Christmas List and not an accounting application, you can choose any wild, cheerful colours you wish for your screen.

The system will now prompt:

```
ENTER BORDER COLOUR?
ENTER BACKGROUND COLOUR?
ENTER CURSOR COLOUR?
```

For each prompt select a colour by pressing 'CTRL' and a number key with a colour name on its side, OR, 'Commodore KEY' and a number key with a colour name on its side

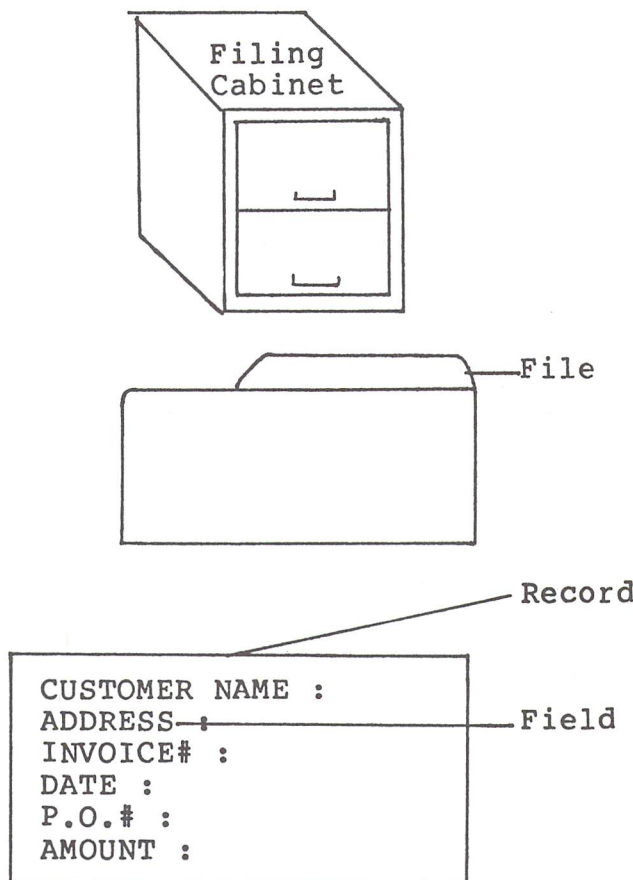
Note that if you chose a black background with black border and black cursor nothing would appear on the screen. This can be remedied by choosing a light cursor colour ie. white or yellow. Refer to the instruction manual that came with your 64 for further details.

Designing A Form For Your Christmas List Data

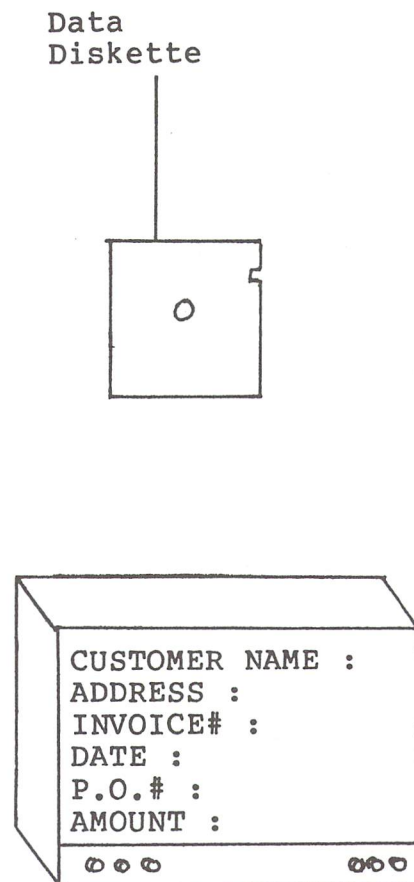
After choosing your screen colours and pressing 'return', your screen will be cleared except for the bottom line where it will say screen '1 of 1' and 'L1 C1'. This means that you are on screen 1 and your cursor is on line 1 column 1. You can now create a screen for your Christmas list.

Picture a file folder filled with blank pieces of paper. The file folder is really our empty data diskette and the blank pieces of paper are blank screens with nothing on them. You are now going to create 'forms' out of the blank pieces of paper.

MANUAL FILING SYSTEM



'64 MANAGER FILING SYSTEM



Picture a 'form' with titles and boxes to fill in. The top of a job application, similar to the one below, may help you visualize a form.

```

JOB APPLICATION FORM
NAME      : _____
ADDRESS   : _____
    
```

Notice that a form has titles or prompts followed by specific areas in which to enter information. We are going to use a form for entering information in our Christmas List.

Using the screen as a blank piece of paper, you will now design a form that accomodates all the information you need to know about a person on your Christmas List. When you are finished it may resemble the screen below. Each '↑' indicates the beginning and end of a box or field of information. The 'π' represents a field that will only need 1 character of information.

```

CHRISTMAS LIST '83
NAME      ↑                ↑ M/F π
ADDRESS   ↑                π
CITY      ↑                ↑
STATE     ↑                ↑
ZIP       ↑                ↑
PHONE     ↑                ↑
BIRTHDAY  ↑↑↑/↑↑/↑↑
LAST CONTACT ↑↑/↑↑/↑↑
NOTES     ↑                ↑

LAST YEAR
REC'D: CARD π    PARTY π    VISIT/CALL π
      GIFT ↑      ↑        $ ↑        ↑
GAVE: CARD π    PARTY π    VISIT/CALL π
      GIFT ↑      ↑        $ ↑        ↑
NOTES ↑

THIS YEAR
CARD π    PARTY INVIT. π    VISIT/CALL π
      GIFT ↑      ↑        $ ↑        ↑
NOTES ↑
    
```

I will start you off with title of the page and the 'NAME' entry and you can design the rest of the form yourself. Cursor over to column 14 line 1, type 'CHRISTMAS LIST '83', then press 'return'. You will now be on line 2 column 1. Press return again to get to row 3. Now cursor over to column 4 (3 spaces) and type in 'NAME'. This is the first title (or screen prompt) for the first field in your form. Now you are going to indicate

the field size. While holding the CONTROL key down, press 'f'. After releasing the keys the prompt 'LENGTH?' will appear on the bottom line. The program will accept numbers between 1 and 30 for the length of the field. In our case, we have chosen 20 characters as the length of the 'name' field. Enter '20' and press 'return'. You will notice that this command will leave two ↑'s to indicate where the field is. The length of the field on the screen includes the two ↑'s. (The ↑'s may also be keyed in directly, but you will have to count the field length yourself. Be sure to include the ↑'s in your field count.)

In a similar manner to the above, fill in the rest of the form repeating the 3 steps below for each piece of information.

1. Decide on a specific piece of information you want included on the form. (Information such as name, address, phone, whether you want to send them a card or gift, etc.)
2. At a suitable spot on the screen, enter a title or prompt for that piece of information to be input, eg. "NAME" prompts the user to input a name. There must be sufficient room on the right side of the prompt to enter the required information.
3. Specify the maximum number of letters or characters that can be input into the field following the title or prompt. This is done by holding the control key down and pressing 'f', entering a number for the field length, then pressing 'return'. (A single character field will show a 'π' sign, fields greater than 1 character are represented by ↑'s at the precise beginning and end of the field.)

Repeat the three steps above again and again until you have all the information you want on the form. After you have entered all the prompts and fields in the form, you may want to pause to consider the overall design of the form. Only you can decide on what is a pleasing layout. It should be clear and easy to read. Revisions may be made now or later using the 'CREATE/REVISE' option. Once you have started entering data in the 'ENTER/EDIT' option, you may NOT change the size of the fields. Our sample will give you an idea of what kinds of information you may want to store and how to lay it out. The numbers stand for the maximum number of letters or characters that may be entered in a field or information box.

Saving The Screen

After you have finished designing the form on the screen and you are sure it is correct, press the 'left arrow' key. The screen will then prompt you with 'ARE YOU SURE?'. Enter

'y' and press 'return'. The program will check to see if all fields are closed (fields longer than 1 character must have a pair of ↑s. If there is an error, you will be prompted to change the field entry.

Next the screen will prompt 'Another Screen?'. In our case we will enter 'n' and press 'return'. Then the screen will prompt 'Do you wish to alter field types?' this is to change between an alpha field ie. letters of the alphabet and numeric fields ie. numbers. As we do not really have any purely numeric fields, we will leave all the fields the same by entering 'n' and 'return'.

Deciding On The Number Of Records For Your File

Now place the data disk you previously formatted into the disk drive after removing the manager disk. The screen will prompt you with 'Rec Len XXXX, Max No of Rec XXXX' at the bottom of the screen and asks 'OK?'. Enter 'y' and 'return'. Then the screen will prompt 'Rec Len, # of records' at which point you can accept the number of records displayed by pressing 'return' or enter your own number of records (up to about 300) and press 'return'. The computer will then go to the disk and create a file large enough to handle all the records you asked for. As this process will take some time, you may want to go to the kitchen for a snack.

Entering Individual Records In Your File

Now you may enter the individual records for people on your Christmas List using the form you have just designed on the screen. Choose 'Enter/Edit' on the main menu. In this option you may enter, change, delete and search for records of individual people in your Christmas List. (More advanced commands will be discussed in future articles.) For each person on your Christmas List, complete a screen form with the following steps:

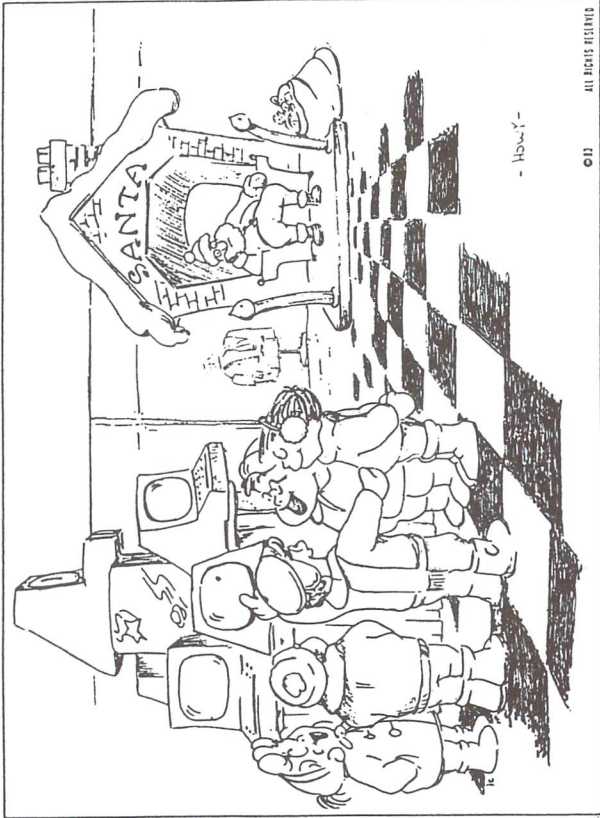
1. Press 'e' to enter a record;
2. Fill in the blank fields, pressing 'return' after each entry;
3. Press ← to save the record. When you have entered a record for every person on your Christmas list, press F1 to return to the main menu. The screen will prompt 'Are you sure (y/n)?', enter 'y', 'return'.

More To Come

In the next article, we will discuss how to do sorts, index searches and generate many useful lists such as a Christmas card list, a gift list, a party list, and a birthday reminder list.

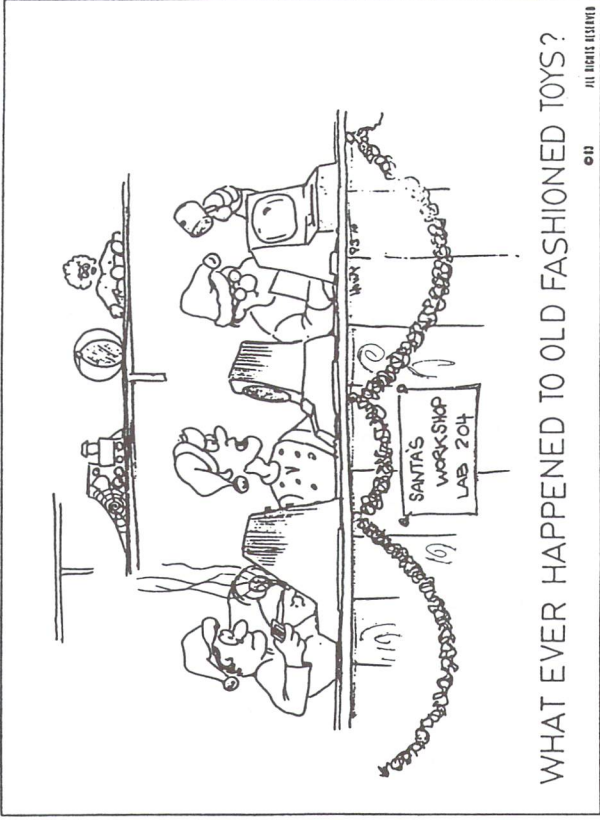
Merry Christmas

from Mike Panning & Howy Parkins



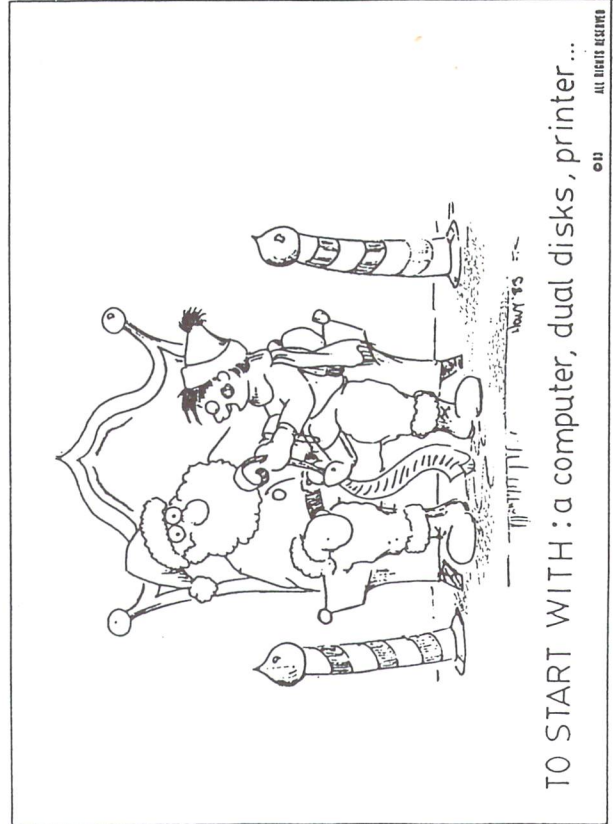
© 1993 ALL RIGHTS RESERVED

- Howy -



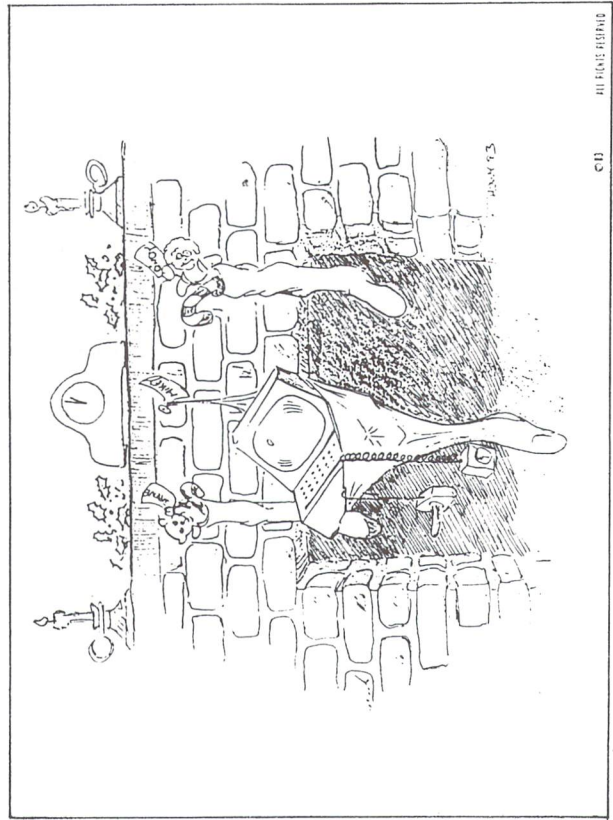
© 1993 ALL RIGHTS RESERVED

WHAT EVER HAPPENED TO OLD FASHIONED TOYS?



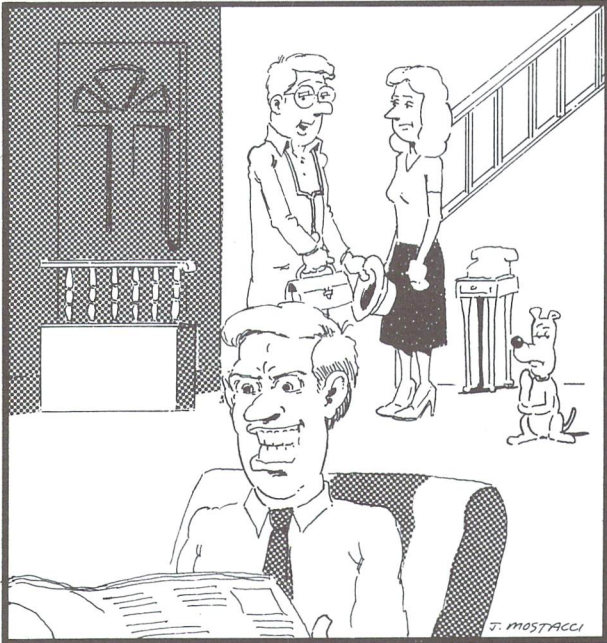
© 1993 ALL RIGHTS RESERVED

TO START WITH : a computer, dual disks , printer ...

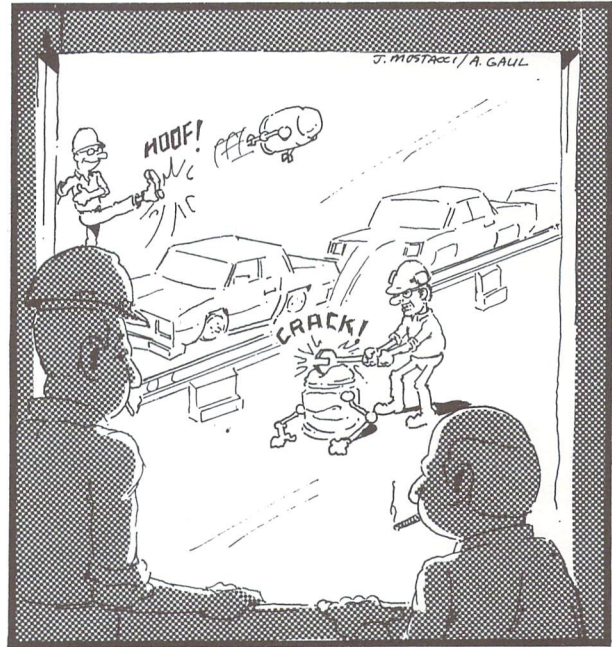


© 1993 ALL RIGHTS RESERVED

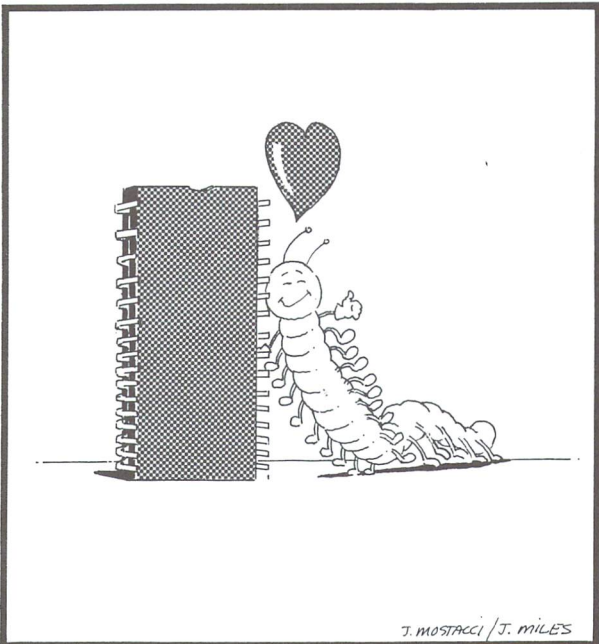
CompuKinks.



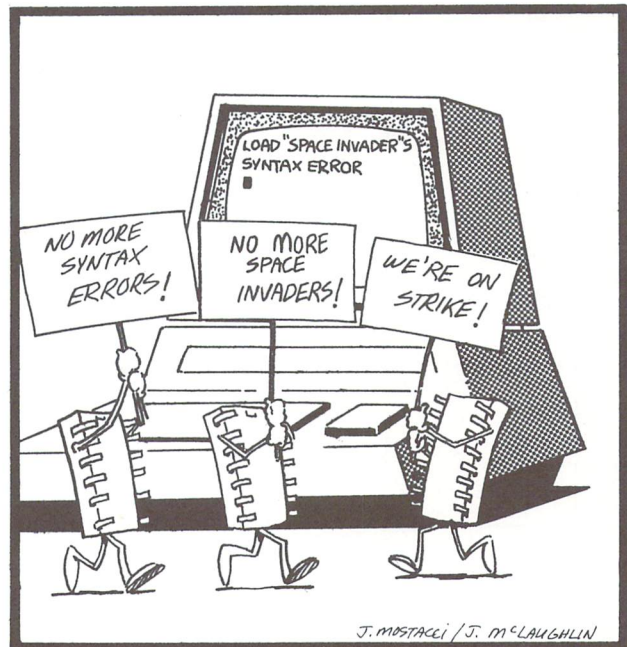
Don't worry Mrs. Smith, his facial muscles will eventually relax. But really, he shouldn't be playing video games for such long periods of time.



I see the Union is laying off more robots



J. MOSTACCI / J. MILES



J. MOSTACCI / J. McLAUGHLIN

BASIC – Beginners All-Purpose Symbolic Instruction Code

Commands, Statements, and Symbols

Command/Statement	Example	Purpose
CLOSE	10 CLOSE n	Closes logical file 'n'.
CLR	CLR	Sets variables to zero or null.
CMD	CMD D	Keep ieee device 'D' open to monitor bus.
CONT	CONT	Continue program execution after a stop command. No program changes are permitted.
DATA	10 DATA 1,2,3,4 20 DATA TOM, SUE 30 DATA "DOE, TOM"	Specifies data to be read left to right. Alphabets do not need to be enclosed in quotes. if strings contain spaces, commas, colons, or graphic characters, the string must be enclosed in quotes.
DEF	10 DEF FN R(X)	Defines function 'R'
DIM	10 DIM A(n) 20 DIM A(n,m,o,p) 30 DIM A(n),B(m) 40 DIM A(N) 50 DIM A\$(n)	Specifies maximum number of elements in an array or matrix. Specifies maximum number of dimensions in an array. Number of arrays limited by memory. May be dimensioned dynamically. Strings to be dimensioned.
END	999 END	Terminates program execution.
FOR	10 FOR I = 1 TO 10	Begins repetitive loop, specifying loop variable and number of intended iterations (in this example 'I' for 10 iterations).
FRE	PRINT FRE (0)	Returns number of bytes of available memory.
GET	10 GET C 20 GET C\$ 30 GET #d, C 40 GET #d, C\$	Accepts single character from keyboard. Accepts single string character from keyboard. Accepts single character from specified logical file. Accepts specified single string character from logical file.
GOSUB	10 GOSUB n	Begins execution of a subroutine which begins on line 'n'.
GOTO	GOTO n	Continue program execution at line n after a stop command. Program changes are permitted.
IF...GOTO	10 IF X = 10 GOTO n	Transfers execution to line 'n' if result of condition is true.
IF...THEN	10 IF X = 10 THEN Y = 3	Code following THEN is executed only if result of condition is true. May also be followed by line number to transfer execution.
INPUT	10 INPUT A 20 INPUT A\$ 30 INPUT A,A\$,B,B\$ 40 INPUT #d, A 50 input #d, a\$ 60 INPUT #d, A,A\$,B,B\$	Accepts value of 'A' from keyboard. Accepts value of string variable 'A' from keyboard. The string does not have to be enclosed in quotes. Accepts specified values from keyboard. Accepts value of 'A' from logical file 'd'. accepts specified string from logical file 'd'. Accepts specified values and string from logical file 'd'. Strings do not have to be enclosed in quotes.
LET	LET X = 10	Optional. Assigns variable 'X' the value of 10.
LIST	LIST LIST -n LIST n-m LIST n-	Lists current program. Lists current program through line 'n'. Lists lines 'n' through 'm' of current program. Lists current program from line 'n' to end.
LOAD	10 LOAD 20 LOAD "NAME" 30 LOAD "NAME", d 30 LOAD "NAME", d, c	Loads next encountered program from tape unit into memory. Loads program or file 'NAME' into memory from tape unit. Loads specified file 'NAME' from device 'd'. Loads specified file 'NAME' from device 'd' for command 'c'. (VIC/C64 only – c = 1 for direct memory load)

Command/ Statement	Example	Purpose
NEW	NEW	Deletes current program in memory, sets variables to zero.
NEXT	NEXT	Indicates end of code contained in a FOR/NEXT loop.
ON...GOSUB	10 ON A GOSUB I, m, n	Begins execution of subroutine which begins on specified line (in this example, 'I', 'm', or 'n') depending on value of index 'A'.
ON...GOTO	10 ON A GOTO I, m, n	Transfers control to specified line 'I', 'm', or 'n' depending on value of index 'A'.
OPEN	10 OPEN a 20 OPEN a, d 30 OPEN a, d, c 40 OPEN a, d, c, "NAME"	Opens logical file 'a' for read only from tape unit. Opens logical file 'a' for read only from device 'd'. Opens logical file 'a' for command 'c' from device 'd'. Opens logical file 'a' on device 'd'. If device 'd' accepts formatted files, file name is positioned for command.
PEEK	PEEK(a) PEEK(A)	Returns byte value from address 'a'. Address can be dynamic.
POKE	POKE a, b POKE A, B	Puts byte 'b' into address 'a'. Parameters can be dynamic.
POS	10 PRINT POS(0)	Prints next available print position (position of cursor on screen).
PRINT	10 PRINT A 20 PRINT A\$ 30 PRINT A, A\$ 40 PRINT #d, A 50 PRINT #d, A\$	Prints value 'A' on display screen. Prints specified string on screen. Prints specified values or strings on screen, beginning in next available print position (pre-tabbed positions are in columns 10,20,30,40 etc.). Prints value of 'A' on device 'd'. Prints specified string on device 'd'.
READ	10 READ A\$, B\$	Reads next two data elements into variables A\$ and B\$.
REM	10 REM comment	Remark indicator. Execution skips entire line.
RESTORE	10 RESTORE	Resets data pointer so that next READ receives first element of first DATA statement.
RETURN	9990 RETURN	Subroutine exit; transfers control to the statement following most recent gosub directing transfer to the subroutine.
RUN	RUN RUN n	Begins execution of program at lowest line number. Begins execution of program a line 'n'.
SAVE	SAVE "NAME" SAVE "NAME", d SAVE "NAME", d, c	Saves current file or program 'NAME' on tape unit. Saves current program or file 'NAME' on device 'd'. Saves file 'NAME' on device 'd'. 'c' specifies eof or eot.
STEP	10 FOR I=1 TO 10 STEP 2	Alters loop variable increment.
STOP	STOP	Stops program execution.
SYS	SYS (x)	Complete control of pet is transferred to a subsystem at decimal address contained in the argument. Brackets optional.
USR	USR (x)	Transfers program control to a program whose address is at locations 1 and 2 (VIC/C64 - locations 784,785). 'x' is a parameter passed to and from the machine language program.
VERIFY	VERIFY VERIFY "NAME" VERIFY "NAME", d	Verifies current program against next program on tape unit. Verifies current program against program 'NAME' on tape unit. Verifies current program 'NAME' on device 'd'.
WAIT	WAIT a, b, c	Halts execution of Basic until contents of address 'a', and 'ed' with value 'b' and exclusive or 'ed' with value 'c', is not equal to zero. 'c' is optional and defaults to zero.

Symbol	Example	Purpose
=	10 A = B	Assigns a value to a variable.
↑	30 PRINT A↑2	Exponentiation
/	40 C = A/8	Division.
*	50 C = A*8	Multiplication.
+	60 C = A + 8	Addition.
-	70 C = A - 8	Subtraction.
=	10 IF A = B THEN PRINT C	'A' Equals 'B'.
<>	10 IF A<>B THEN C = 4	'A' Does not equal 'B'.
<	10 IF A<B THEN C\$ = " X "	'A' Is less than 'B'.
>	10 IF A>B THEN C\$ = " Y "	'A' Is greater than 'B'.
<=	10 IF A<= B THEN C = 20	'A' Is less than or equal to 'B'.
>=	10 IF A>= B THEN C = D-1	'A' Is greater than or equal to 'B'.
AND	10 IF A AND B THEN C = 9	'A' and 'B' must both be true for statement 10 to be true.
OR	20 IF A OR B THEN C = 9	'A' must be true or 'B' must be true for statement 20 to be true.
NOT	30 IF NOT A THEN PRINT C	Expression is true if 'A' is false.

Note: the numerical values used in the evaluation of logical comparisons are: 'true' is any non-zero number and 'false' is zero.

Special Symbols

Symbols	Example	Purpose
:	10 A = 1:B = 2:C = 3	Allows multiple statements on a line.
;	10 PRINT A;B 20 PRINT A\$;B\$	Allows same line printing. Elements are separated by 3 spaces. Allows same line printing. String elements are concatenated.
.	X = 10.99	Decimal point
,	10 PRINT A, B LOAD " NAME " ,d	Allows same line printing. Elements are separated and printed in pre-tab'ed print positions (columns 10,20,30, etc.). Separates parameters in load, save, open, mid\$, on..goto, etc.
?	10 ?A	Abbreviation for 'print'. Stores as one character; lists as word PRINT.
\$	10 A\$ = " ABCDEFG "	String identifier.
%	10 A% = INT(X)	Integer identifier.
#	10 PRINT#8	Logical file number indicator
"	10 A\$ = " ABCDEFG "	String enclosures.
()	X = (A-2)/(B + 2)	Expression priority evaluation
π	10 C = π*D	Value of Pi 3.1415927.

Reserved Variables

Variable	Purpose
DS	Disk Status number (except 2.0)
DS\$	Disk Status string (except 2.0)
EL	Error Line (B Series only)
ER	Error number (B only)
ERR\$()	Error String array. See table for messages. (B only)
TI	Time in Jiffies (1/60th's sec.) since power up or TI\$ reset (except B Series)
TI\$	Time in HHMMSS
ST	The Status variable. See table for functions.

Hierarchy of Operations

Operator	Description
()	Brackets always dictate priority
↑	Exponentiation
-	Negation (unary minus)
* /	Multiplication & Division
+ -	Addition & Subtraction
< = >	Relational Operations
NOT	Logical NOT (Integer two's complement)
AND	Logical AND
OR	Logical OR

Basic 4.0 Disk Commands

Function	Example	Purpose
APPEND	10 APPEND#d, "NAME"	Open file 'NAME' on device 'd' for appending. New data is added to end of existing data.
BACKUP	BACKUP D0 TO D1	Duplicate disk in drive 0 onto disk in drive 1
CATALOG	CATALOG D0	Displays list of filenames in specified drive.
COLLECT	COLLECT D1	Purges disk in specified drive of any improperly closed files (indicated by * beside file type).
CONCAT	CONCAT "NAME1" TO "NAME2", D1	Concatenates file "NAME1" to file "NAME2". I.e. NAME2 = NAME2 + NAME1
COPY	COPY "NAME", D0 TO "NAME", D1 COPY "NAME", D0 TO "DUP", D0 COPY D0 TO D1	Copies file "NAME" from drive 0 to drive 1 Makes duplicate of file "NAME" Copies entire contents from D0 to D1
DCLOSE	DCLOSE #n	Closes disk logical file 'n'
DIRECTORY	DIRECTORY D0	Exact same as Catalog. Use preference.
DLOAD	DLOAD "NAME", Dd,Uu	Loads program "NAME" from drive 'd' on unit 'u'
DOPEN	DOPEN#n, "NAME", Dd,Uu DOPEN#n, "NAME", Dd,Uu,W	Opens file "NAME" for reading from drive 'd', unit 'u'. Default values: d = 0, u = 8. Data is retrieved through file number 'n'. Opens file "NAME" for writing to drive 'd', unit 'u'. Not necessary for RELative files.
DSAVE	DSAVE "NAME", Dd,Uu	Saves current program to drive 'd' on unit 'u' as file "NAME"
HEADER	HEADER "DISKNAME", Dd,lid,Uu	Formats disk in drive 'd' unit 'u' assigning it a "DISKNAME" and 'id'.
RECORD	10 RECORD#n, a	Positions relative file open on logical file number 'n' to record number 'a'. 'a' may be dynamic but must be enclosed in brackets.
RENAME	RENAME "NAME" TO "NEWNAME", D0	Changes a file name.
SCRATCH	SCRATCH "NAME", D1	Eliminates file "NAME" from disk.

Additional B Series Commands

Function	Example	Purpose
BANK	BANK b	Sets bank number to 'b'.
BLOAD	BLOAD "NAME", Dd,Uu,ON Bb,Pp	Loads file "NAME" from drive 'd' unit 'u' into bank 'b' at position 'p'
BSAVE	BSAVE "NAME" ON Bb,Pp1 to Pp2	Saves current memory in bank 'b' from address 'p1' to 'p2' as file "NAME" to drive 0 unit 8. Addresses are in decimal.
DCLEAR	DCLEAR D1	Initialize disk in drive 1
DELETE	DELETE 10-30	Deletes lines from current program. Specify line range same as LIST.
DISPOSE	DISPOSE GOSUB	Purges stack of unwanted return addresses (like 'POP')
ELSE	IF ST THEN E = 1 ELSE E = 0	Alternate condition following IF..THEN. May also be used to transfer execution
INSTR	PRINT INSTR (A\$, B\$)	Returns position of string B\$ within A\$. Returns 0 if not found.
KEY	KEY KEYn, "CATALOG D0" + CHR\$(13)	Displays list of function key definitions Defines function key 'n'.
PUDEF	PUDEF "-.,£"	Re-defines Print Using format characters. Default is " ,,\$". In this example, space is changed to '-', comma to period, period to comma, and dollars to pounds.
RESUME	RESUME RESUME n RESUME NEXT	Continues execution after program error or editing Resumes execution at line 'n' Resumes execution at start of current active FOR/NEXT
TRAP	TRAP 50000	Specifies routine at line 50000 as an ON ERROR routine.
USING	PRINT USING "-\$##,###";X	Specifies format to be used for numerical output.

Function	Example	Purpose
ASC	10 A = ASC("XYZ")	Returns the integer value corresponding to ASCII code of the first character in string.
CHR\$	10 A\$ = CHR\$(n)	Returns character corresponding to ASCII code number.
LEFT\$	10 PRINT LEFT\$(X\$, a)	Returns leftmost 'a' characters from string.
LEN	10 PRINT LEN(X\$)	Returns length of string.
MID\$	10 PRINT MID\$(X\$, a, b)	Returns 'b' characters from string, starting with the 'a'th character.
RIGHT\$	10 PRINT RIGHT\$(X\$, a)	Returns rightmost 'a' characters from string.
STR\$	10 A\$ = STR\$(A)	Returns string representation of variable 'A'
VAL	10 A = VAL(A\$) 20 A = VAL(" A ")	Returns numeric representation of string. If string not numeric, returns "0".

ASC, LEN and VAL functions return numeric results. They must be used as part of any expression. Assignment statements are used here for examples only; other statement types may be used.

Arithmetic Functions

Function	Example	Purpose
ABS	10 C = ABS(A)	Returns magnitude of argument without regard to sign.
ATN	10 C = ATN(A)	Returns arctangent of argument. 'c' will be expressed in radians.
COS	10 C = COS(A)	Returns cosine of argument. 'A' must be expressed in radians.
DEF FN	10 DEF FNA(B) = C * D	Allows user to define a function. Function label 'a' must be a single letter; argument 'b' is a dummy.
EXP	10 C = EXP(A)	Returns constant 'e' raised to the power of the argument.
INT	10 C = INT(A)	Returns largest integer less than or equal to argument.
LOG	10 C = LOG(A)	Returns natural logarithm of argument. Argument must be greater than or equal to zero.
RND	10 C = RND(A)	Generates a random number between zero and one. If 'a' is less than 0, the same random number is produced in each call to rnd. If 'a' = 0, the same sequence of random number is generated each time rnd is called. If 'a' is greater than 0, a new sequence is produced for each call to rnd.
SGN	10 C = SGN(A)	Returns -1 if argument is negative, returns 0 if argument is zero, and returns +1 if argument is positive.
SIN	10 C = SIN(A)	Returns sin of argument. 'A' must be expressed in radians.
SQR	10 C = SQR(A)	Returns the square root of argument.
TAN	10 C = TAN(A)	Returns tangent of argument. 'A' must be expressed in radians.

Mathematical Functions

Function	Basic Equivalent
Secant Cosecant Cotangent	SEC(X) = 1 / COS(X) CSC(X) = 1 / SIN(X) COT(X) = 1 / TAN(X)
Inverse Sine Inverse Cosine Inverse Secant Inverse Cosecant Inverse Cotangent	ARCSIN(X) = ATN(X / SQR(-X*X + 1)) ARCCOS(X) = ATN(X / SQR(-X*X + 1)) + π/2 ARCSEC(X) = ATN(X / SQR(X*X - 1)) ARCCSC(X) = ATN(X / SQR(X*X - 1)) + (SGN(X) - 1 * π/2) ARCCOT(X) = ATN(X) + π/2
Hyperbolic Sine Hyperbolic Cosine Hyperbolic Tangent Hyperbolic Secant Hyperbolic Cosecant Hyperbolic Cotangent	SINH(X) = (EXP(X) - EXP(-X)) / 2 COSH(X) = (EXP(X) + EXP(-X)) / 2 TANH(X) = EXP(-X) / (EXP(X) + EXP(-X)) * 2 + 1 SECH(X) = 2 / (EXP(X) + EXP(-X)) CSCH(X) = 2 / (EXP(X) - EXP(-X)) COTH(X) = EXP(-X) / (EXP(X) - EXP(-X)) * 2 + 1
Inverse Hyperbolic Sine Inverse Hyperbolic Cosine Inverse Hyperbolic Tangent Inverse Hyperbolic Secant Inverse Hyperbolic Cosecant Inverse Hyperbolic Cotangent	ARCSINH(X) = LOG(X + SQR(X*X + 1)) ARCCOSH(X) = LOG(X / SQR(X*X - 1)) ARCTANH(X) = LOG((1 + X) / (1 - X)) / 2 ARCSECH(X) = LOG(SQR(-X*X + 1) + 1/X) ARCCSCH(X) = LOG(X / SQR(X*X - 1)) + (SGN(X) - 1 * π/2) ARCCOTH(X) = LOG(X) + π/2

ST - The Status Variable

Bit	Val	Cassette Read	IEEE	Tape Load/Ver.	Vic/64 RS-232
0-7	0	OK	OK	OK	OK
0	1		time out on write		parity error
1	2		time out on read		framing error
2	4	short block		short block	rec. buffer overrun
3	8	long block		long block	unused
4	16	unrecoverable read error		any mismatch	CTS signal missing
5	32	checksum error		checksum error	unused
6	64	end of file	EOI		DSR signal missing
7	-128	end of tape	device not present	end of tape	break detected

Printer Control Characters

CHR\$ values are sent to print file with Secondary Addr 0 or 1

CHR\$	Operation
10	Line Feed
13	'Carriage Return' (automatic Line Feed on CBM printers)
14	Begin double-width character mode
15	End double-width character mode
18	Begin reverse character mode
146	End reverse character mode
19	Set top of page
147	Feed to top of next page
17	Switch to upper/lower case character set
145	Switch to upper case/graphics character set
16	Tab to position in next 2 characters
27	Move to specified dot position
8	Begin dot-programmable graphic mode
26	Repeat graphics data

Table Of Secondary Addresses

Eg. OPEN 4, 4, 7 ; 7 is the Secondary Address on CBM printers that alters line spacing. Once open the new value can be sent. Secondary addresses are not applicable to the VIC 20/Commodore 64 RS-232 routines ('device' 2), keyboard (device 0), screen (device 3), or the CBM 8010 Modem (device 5).

Sec. Addr.	Printer 4	I/O Device & Device Number (DV#)		
		Cassette 1 or 2	Vic/64 Cassette 1	Disk 8
0	Print data exactly as received	seq. read	Load & relocate (dflt)	Load, and Dir read
1	Print data according to previously defined format	Write file + end-of-file marker on Close	Load without relocating	Program Save
2	Format Set-up	Write file + eof + end of tape marker on Close	Write file + eof + end of tape marker on Close	R/W channels are 2-14
3	Set number of lines per page for paging			
4	Enable printer format diagnostics			
5	Define a programmable character			
6	Set spacing between lines			
7	Upper/Lower case			
8	ASCII/Graphics			
9	Suppress Diagnostic Message Printing			
10	Reset Printer			
11	Set Uni-Direction			
12	Reset Uni-Direction			
13	Set Condense mode			
14	Reset Condense mode			
15	Set pseudo letter quality			
21	Reset pseudo letter quality			
17	Storing bit image data			
18	Printing bit data previously written			
Command Ch.				

DS	Error Description
0	OK, no error exists
1	files scratched response (not an error)
2-19	Unused: can occur, should be ignored
20	read error; block header not found
21	read error; sync character not found
22	read error; data block not present
23	read error; checksum error in data
24	read error; byte decoding error
25	write error; write verify error
26	write protect on
27	read error; checksum error in header
28	write error; data extends into next block
29	disk id mismatch
30	syntax error; general syntax
31	syntax error; invalid command
32	syntax error; command line > 58 chars
33	syntax error; invalid filename
34	syntax error; no filename given
39	syntax error; command file not given
50	record not present
51	overflow in record
52	file too large
60	file open for write
61	file not open
62	file not found
63	file exists
64	file type mismatch
65	no block; t,s is next available block
66	illegal track or sector
67	illegal system track or sector
70	no channels (available)
71	dir error (directory error)
72	disk full or directory full
73	cbm dos v2 (or v2.x for later dos's); power up message, also indicates write attempt with dos mismatch
74	drive not ready
75	format speed error
76	controller error

8032 Control Characters

Most functions can be activated by combinations of simultaneous key depressions, a phenomena of the keyboard hardware. Notice that the CHR\$ values of complimentary functions differ by 128.

Function	CHR\$	ESC/RVS	Keyboard Combination
BELL	7	G	
GRAPHICS TEXT	142 14	Shift N N	Both Shifts + "
SCROLL DOWN SCROLL UP	153 25	Shift Y Y	Left Shift + TAB + I
SET BOTTOM SET TOP	143 15	Shift O O	Shift + Z + A + L Z + A + L
INSERT LINE DELETE LINE	149 21	Shift U U	Shift + RVS + A + L RVS + A + L
ERASE BEGIN ERASE END	150 22	Shift V V	Shift + TAB + \boxtimes + DEL TAB + \boxtimes + DEL
SET/CLR TAB TAB	137 9	Shift I I	Shift + TAB TAB

8032 Window POKEs	
TOP:224, T where T=0 to 24	LEFT:226, L where L=0 to 79
BOTTOM:225, B where B=T to 24	RIGHT:213, R where R=L to 79

Error Messages

Message	Description
BAD DATA	String data was received from an open file, but the program was expecting numeric data.
BAD SUBSCRIPT	The program was trying to reference an element of an array whose number is outside of the range specified in the DIM statement.
CAN'T CONTINUE	The CONT command will not work, either because the program was never 'RUN', there has been an error, or a line has been edited.
DEVICE NOT PRESENT	The required I/O device was not available for an 'OPEN', 'CLOSE', 'CMD', 'PRINT#', 'INPUT#', or 'GET#'. Division by zero is a mathematical oddity and not allowed.
DIVISION BY ZERO	Division by zero is a mathematical oddity and not allowed.
EXTRA IGNORED	Too many items of data were typed in response to an input statement. Only the first few items were accepted.
FILE NOT FOUND	If you were looking for a file on tape, an 'end-of-tape' marker was found. If you were looking on a disk, no file with that name exists.
FILE NOT OPEN	The file specified in a 'CLOSE', 'CMD', 'PRINT#', 'INPUT#', or 'GET#', must first be 'OPEN'ed.
FILE OPEN	An attempt was made to OPEN a file using the number of an already open file.
FORMULA TOO COMPLEX	The string expression being evaluated should be split into at least two parts for the system to work with, or a formula has too many parentheses.
ILLEGAL DIRECT	The 'INPUT' statement can only be used within a program, and not in direct mode.
ILLEGAL QUANTITY	A number used as the argument of a function or statement is out of the allowable range.
LOAD	A problem has occurred during program LOAD, disk or tape
NEXT WITHOUT FOR	This is caused by either incorrectly nesting loops or having a variable name in a 'NEXT' statement that doesn't correspond with one in a 'FOR' statement.
NOT INPUT FILE	An attempt was made to 'INPUT' or 'GET' data from a file which was specified to be for output only.
NOT OUTPUT FILE	An attempt was made to 'PRINT' data to a file which was specified as input only.
OUT OF DATA	A 'READ' statement was executed but there is no data left unread in a 'DATA' statement.
OUT OF MEMORY	There is no more 'ram' available for program or variables. This may also occur when too many 'FOR' loops have been nested, or when there are too many 'GOSUB's in effect.
OVERFLOW	The result of a computation is larger than the largest number allowed, which is 1.70141884e + 38.
REDIM'D ARRAY	An array may only be 'DIM' ensioned once. If an array variable is used before that array is 'DIM'd, an automatic 'DIM' operation is performed on that array setting the number of elements to ten, and any subsequent 'DIM's will cause this error.
REDO FROM START	Character data was typed in during an 'INPUT' statement when numeric data was expected. Just re-type the entry so that it is correct, and the program will continue by itself.
RETURN WITHOUT GOSUB	A 'RETURN' statement was encountered, and no 'GOSUB' command has been issued.
STRING TOO LONG	(except 2.0) A string can contain up to 255 characters.
SYNTAX	A statement or command is unrecognizable. A missing or extra parenthesis, misspelled keywords, etc.
TYPE MISMATCH	This error occurs when a number is used in place of a string, or vice-versa.
UNDEF'D FUNCTION	A user defined function was referenced, but it has never been defined using the 'DEF FN' statement.
UNDEF'D STATEMENT	An attempt was made to 'GOTO' or 'GOSUB' or 'RUN' a line number that doesn't exist.
VERIFY	The program on tape or disk does not match the program currently in memory.

B-Series Error Messages

This list is a summary of B-Series error messages that are displayed by PRINTING ERR\$(X) where X equals the value down the left column.

X	Message	Explanation
0	?STOP KEY DETECTED	Occurs when doing a KERNAL I/O function and the STOP key is pressed. May occur during LOAD or SAVE (or OPEN, CLOSE, GET#, INPUT#, PRINT# when the cassette tape is moving). CLOSE any open write files to save data.
1	?TOO MANY FILES	Maximum OPEN files is ten.
2	?FILE OPEN	An attempt was made to OPEN or DOPEN a file with a file number already in use.
3	?FILE NOT OPEN	An attempt was made to access a file not previously OPEN or DOPENed
4	?FILE NOT FOUND	The file specified in OPEN or LOAD was not found on the device specified. In the case of tape I/O, and end of tape marker was encountered.
5	?DEVICE NOT PRESENT	An attempt was made to access a device not currently connected or powered-up on the IEEE-488 bus. May happen on OPEN, CLOSE, CMD, INPUT#, GET#, PRINT#. If filename is not specified with OPEN, this error will occur.
6	?NOT INPUT FILE	An attempts was made to read a file originally OPENed for writing.
7	?NOT OUTPUT FILE	An attempts was made to write data to a file originally OPENed for reading. The keyboard cannot be written to.
8	?MISSING FILENAME	All LOADs and SAVEs from the IEEE port (eg. disk) require a filename.

X	Message	Explanation
9	?ILLEGAL DEVICE NUMBER	Occurs if you try to access a device in an illegal manner. For example, LOADing or SAVEing from/to the keyboard, screen, or RS-232.
10	?ARE YOU SURE	Confirmation prompt for BACKUP, SCRATCH, and HEADER. It is not an error message and occurs only in direct mode, not during BASIC program execution.
11	?BAD DISK	Media failure on HEADER command.
12	<return> READY. <return>	This Is Not An Error Message. This message lets you know that your system is ready to use.
13	<space> IN <space>	Not An Error Message. Used to indicate which line an error has occurred "in".
14	?BREAK	This occurs when the STOP key is pressed during BASIC execution. CONT can be used to restart the program.
15	?EXTRA IGNORED	Too many items of data or separators were entered in response to an INPUT statement.
16	?REDO FROM START	This diagnostic message occurs when a numeric variable is used with INPUT and non-numeric data is received. INPUT continues to function until acceptable data has been received.
17	Last Evaluated Number	This Is Not An Error Message. This is the last value that has been processed through the numerical output buffer. (eg. print 100/10 : print ERR\$(17) ...will print 10 both times.
18	"MORE" <return>	This Is Not An Error Message. Prints "MORE" and carriage return.
19	Power On Message	This Is Not An Error Message. Prints: *** COMMODORE BASIC 128, B4.0*** *** COMMODORE BASIC 256, B4.0***
20	?NEXT WITHOUT FOR	Either a NEXT is improperly nested or the variable in a NEXT statement corresponds to no previously executed FOR statement.
21	?SYNTAX	BASIC cannot recognize the statement you have typed. Caused by such things as missing parenthesis, illegal characters, incorrect punctuation, misspelled keyword.
22	?RETURN WITHOUT GOSUB	A RETURN statement was encountered with noprevious GOSUB.
23	?OUT OF DATA	An attempt was made to READ data from a DATA statement but no data exists or the program has already read them all.
24	?ILLEGAL QUANTITY	Occurs when a function is accessed with a parameter out of range caused by: 1. A matrix subscript out of range (0 < X < 32767) 2. LOG (negative or zero argument) 3. SQR (negative argument) 4. ATB where A<0 and B not integer. 5. Call of USR before a machine language subroutine has been patched in. 6. Use of string functions MID\$, LEFT\$, RIGHT\$, with length parameters out of range. 7. Index on...GOTO out of range. 8. Addressof PEEK, POKE, WAIT or SYS out of range. 9. Byte parameters of WAIT, POKE, TAB and SPC out of range.
25	?OVERFLOW	Numbers resulting from computations or input that are greater than 1.70141184E + 38 or less than 2.93873587E-39.
26	?OUT OF MEMORY	BASIC text space, or Variables space, or Arrays memory space has been completely filled
27	?UNDEFINED STATEMENT	A GOTO, GOSUB, or THEN has been executed with a line number that does not exist.
28	?BAD SUBSCRIPT	An attempt was made to reference an array element which is outside the dimensions specified in the DIM statement.
29	?REDIM'D ARRAY	An attempt was made to define an array using a variable already used in an array.
30	?DIVISION BY ZERO	Illegal divide. Message is followed by the line number - list and check variables.
31	?ILLEGAL DIRECT	INPUT, INPUT#, GET, GET#, and DEF cannot be used in direct mode.
32	?TYPE MISMATCH	An arithmetic operation has been given non-numeric data, or a string operation has been numeric data.
33	?STRING TOO LONG	Maximum string length is 255 characters. This error will also occur if INPUT# receives more than 80 characters without a carriage return (ie. BASIC input buffer is 80 bytes long).
34	?FILE DATA	Occurs when a numeric variable is used with INPUT# and non-numeric data is received.
35	?FORMULA TOO COMPLEX	BASIC has run out of temporary pointers to keep track of substrings in evaluating a string expression. Break the expression into two smaller parts to cure the problem.
37	?UNDEFINED FUNCTION	Reference was made to a user defined function which had never been defined with DEF.
38	?LOAD ERROR	Cassette tape only. To improve tape reliability, programs are recorded twice with SAVE. This error will occur if LOAD finds recording errors in corresponding positions of both recordings. If more than 31 errors are detected in the first pass, LOAD will not attempt to read the second.
39	?VERIFY ERROR	A VERIFY operation did not match the contents of file with the contents of memory. Re-SAVE your program on another disk or tape.
40	?OUT OF STACK	Too many open FOR...NEXT loops or too many GOSUB calls.
41	?UNABLE TO RESUME	Resume will not operate after a fatal error.
42	?UNABLE TO DISPOSE	All of the DISPOSE type items have been disposed of or none exist.
43	?OUT OF TEXT	A LOAD or DLOAD has attempted to bring in a file larger than 64K. This error will not occur when using the BLOAD command.

Sprite Design

Sprite Colour #2 _____: POKE 53285, _____
 Sprite Colour #3 _____: POKE 53286, _____
 Sprite Enable: POKE 53269, PEEK(53269) OR 2 ↑ Sprite#
 POKE Sprite X-Expand: POKE 53264, PEEK(53264) OR 2 ↑ Sprite#
 Sprite Y-Expand: POKE 53271, PEEK(53271) OR 2 ↑ Sprite#
 Background Priority: POKE 53275, PEEK(53275) OR 2 ↑ Sprite#

Sprite Multi Colour Mode: POKE 53276, PEEK(53276) OR 2 ↑ Sprite#
Multi Colour Mode Bit Pairs
 Background Colour, PEEK(53281), Use: 00
 Sprite Colour Use: 01
 Sprite Colour #2 Use: 10
 Sprite Colour #3 Use: 11

Column	Bit	Bit	Bit	Column		
1	2	3	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0		
0	21	42		00	15	2A
1	22	43		01	16	2B
2	23	44		02	17	2C
3	24	45		03	18	2D
4	25	46		04	19	2E
5	26	47		05	1A	2F
6	27	48		06	1B	30
7	28	49		07	1C	31
8	29	50		08	1D	32
9	30	51		09	1E	33
10	31	52		10	1F	34
11	32	53		11	20	35
12	33	54		12	21	36
13	34	55		13	22	37
14	35	56		14	23	38
15	36	57		15	24	39
16	37	58		16	25	3A
17	38	59		17	26	3B
18	39	60		18	27	3C
19	40	61		19	28	3D
20	41	62		20	29	3E

7 6 5 4 3 2 1 0 | 7 6 5 4 3 2 1 0 | 7 6 5 4 3 2 1 0

Sprite # _____ (0-7)
 Pointer: POKE 2040 + Sprite#, _____
 Sprite Colour: _____: POKE 53287 + Sprite#, _____
 X-Position: POKE 53248 + Sprite#, X Position
 Y-Position: POKE 53249 + Sprite#, Y Position

Column	Bit	Bit	Bit	Column		
1	2	3	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0		
0	21	42		00	15	2A
1	22	43		01	16	2B
2	23	44		02	17	2C
3	24	45		03	18	2D
4	25	46		04	19	2E
5	26	47		05	1A	2F
6	27	48		06	1B	30
7	28	49		07	1C	31
8	29	50		08	1D	32
9	30	51		09	1E	33
10	31	52		10	1F	34
11	32	53		11	20	35
12	33	54		12	21	36
13	34	55		13	22	37
14	35	56		14	23	38
15	36	57		15	24	39
16	37	58		16	25	3A
17	38	59		17	26	3B
18	39	60		18	27	3C
19	40	61		19	28	3D
20	41	62		20	29	3E

7 6 5 4 3 2 1 0 | 7 6 5 4 3 2 1 0 | 7 6 5 4 3 2 1 0

Sprite # _____ (0-7)
 Pointer: POKE 2040 + Sprite#, _____
 Sprite Colour: _____: POKE 53287 + Sprite#, _____
 X-Position: POKE 53248 + Sprite#, X Position
 Y-Position: POKE 53249 + Sprite#, Y Position

Column	Bit	Bit	Bit	Column		
1	2	3	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0		
0	21	42		00	15	2A
1	22	43		01	16	2B
2	23	44		02	17	2C
3	24	45		03	18	2D
4	25	46		04	19	2E
5	26	47		05	1A	2F
6	27	48		06	1B	30
7	28	49		07	1C	31
8	29	50		08	1D	32
9	30	51		09	1E	33
10	31	52		10	1F	34
11	32	53		11	20	35
12	33	54		12	21	36
13	34	55		13	22	37
14	35	56		14	23	38
15	36	57		15	24	39
16	37	58		16	25	3A
17	38	59		17	26	3B
18	39	60		18	27	3C
19	40	61		19	28	3D
20	41	62		20	29	3E

7 6 5 4 3 2 1 0 | 7 6 5 4 3 2 1 0 | 7 6 5 4 3 2 1 0

Sprite # _____ (0-7)
 Pointer: POKE 2040 + Sprite#, _____
 Sprite Colour: _____: POKE 53287 + Sprite#, _____
 X-Position: POKE 53248 + Sprite#, X Position
 Y-Position: POKE 53249 + Sprite#, Y Position

Column	Bit	Bit	Bit	Column		
1	2	3	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0		
0	21	42		00	15	2A
1	22	43		01	16	2B
2	23	44		02	17	2C
3	24	45		03	18	2D
4	25	46		04	19	2E
5	26	47		05	1A	2F
6	27	48		06	1B	30
7	28	49		07	1C	31
8	29	50		08	1D	32
9	30	51		09	1E	33
10	31	52		10	1F	34
11	32	53		11	20	35
12	33	54		12	21	36
13	34	55		13	22	37
14	35	56		14	23	38
15	36	57		15	24	39
16	37	58		16	25	3A
17	38	59		17	26	3B
18	39	60		18	27	3C
19	40	61		19	28	3D
20	41	62		20	29	3E

7 6 5 4 3 2 1 0 | 7 6 5 4 3 2 1 0 | 7 6 5 4 3 2 1 0

Sprite # _____ (0-7)
 Pointer: POKE 2040 + Sprite#, _____
 Sprite Colour: _____: POKE 53287 + Sprite#, _____
 X-Position: POKE 53248 + Sprite#, X Position
 Y-Position: POKE 53249 + Sprite#, Y Position

VIC 20 Screen Memory

To move the screen: POKE 36869, (PEEK(36869) AND 15) OR X
POKE 36866, (PEEK(36866) AND 127) OR Y

X	Y	4*(PEEK(36866) AND 128) + 64*(PEEK(36869) AND 112) = Location	
		Decimal (1/2K blocks)	Hexadecimal
128	0	0	\$0000
128	128	512	\$0200
129	0	1024	0400
129	128	1536	0600
130	0	2048	0800
130	128	2560	0A00
131	0	3072	0C00
131	128	3584	0E00
132	0	4096	1000 (df1t w/exp)
132	128	4608	1200
133	0	5120	1400
133	128	5632	1600
134	0	6144	1800
134	128	6656	1A00
135	0	7168	1C00
135	128	7680	1E00 (default)
136	0	8192	2000
136	128	8704	2200
137	0	9216	2400
137	128	9728	2600
138	0	10240	2800
138	128	10752	2A00
139	0	11264	2C00
139	128	11776	2E00
140	0	12288	3000
140	128	12800	3200
141	0	13312	3400
141	128	13824	3600
142	0	14336	3800
142	128	14848	3A00
143	0	15360	3C00
143	128	15872	3E00

Commodore 64 Screen Memory

To move the screen: POKE 53272, (PEEK(53272)AND15) OR X

X	(3-PEEK(56576) AND 3) * 16384 + (X*64) = Location For Screen at Bank 0 (default):	
	Decimal	Hexadecimal
0	0	\$0000
16	1024	0400 (default)
32	2048	0800
48	3072	0C00
64	4096	1000
80	5120	1400
96	6144	1800
112	7168	1C00
128	8192	2000
144	9216	2400
160	10240	2800
176	11264	2C00
192	12288	3000
208	13312	3400
224	14336	3800
240	15360	3C00

Commodore 64 VIC II Address

To move VIC II: POKE 53272, (PEEK(53272) AND 252) OR X ;X = 3-Bank#

Bank	X	VIC II Chip Address Range	
		Decimal (16K blocks)	Hexadecimal
0	3	0-16383	\$0000-3FFF (default)
1	2	16384-32767	4000-7FFF
2	1	32768-49151	8000-BFFF
3	0	49152-65535	C000-FFFF

Note: Character ROM only available with VIC II in bank 0 or 2

VIC 20 Character Base

To move the character base: POKE 36869, (PEEK(36869) AND 240) OR X

X*	32768 + (PEEK(36869) AND 15) * 1024 = Location	
	Decimal (1K blocks)	Hexadecimal
0	32768-34815	\$8000-87FF (df1t)
1	33792-35839	8400-8BFF
2	34816-36863	8800-8FFF
3	35840-37887	8C00-93FF
4	36864-38911	9000-97FF
5	37888-39935	9400-9BFF
6	38912-40959	9800-9FFF
7	39936-41983	9C00-A3FF
8	0-2047	0000-07FF
9	1024-3071	0400-0BFF
10	2048-4095	0800-0FFF
11	3072-5019	0C00-13FF
12	4096-6143	1000-17FF
13	5020-7167	1400-1BFF
14	6144-8191	1800-1FFF
15	7168-9216	1C00-23FF

* X = PEEK(36869) AND 15

Commodore 64 Character Base

To move the character base: POKE 53272, (PEEK(53272) AND 240) OR X

X*	(3-PEEK(56576) AND 3) * 16384 + (X*64) = Location For Screen at Bank 0 (default):	
	Decimal (2K blocks)	Hexadecimal
0	0-2047	\$0000-07FF
2	2048-4095	0800-0FFF
4	4096-6143	1000-17FF *1
6	6144-8191	1800-1FFF *2
8	8192-10293	2000-27FF
10	10240-12287	2800-2FFF
12	12288-14335	3000-37FF
14	14336-16383	3800-3FFF

* - X = PEEK(53272) AND 14

*1 - Lower 2K of Character ROM (Bank 0 or 2 only) (default)

*2 - Upper 2K of Character ROM (Bank 0 or 2 only)

Character ROM Contents

Character ROM is the same in all machines, but only addressable in VIC 20/C64

2K Block	VIC 20		Commodore 64			Contents
	Default Address		Default Address		VIC II Image	
	Dec (1/2K blocks)	Hex	Dec (1/2K blocks)	Hex	Hex	
0	32768-33279	8000-81FF	53248-53759	D000-D1FF	1000-11FF	Upper case characters
	33280-33791	8200-83FF	53760-54271	D200-D3FF	1200-13FF	Graphics characters
	33792-34303	8400-85FF	54272-54783	D400-D5FF	1400-15FF	Reversed upper case characters
1	34304-34815	8600-87FF	54784-55295	D600-D7FF	1600-17FF	Reversed graphics characters
	34816-35327	8800-89FF	55296-55807	D800-D9FF	1800-19FF	Lower case characters
	35328-35839	8A00-8BFF	55808-56319	DA00-DBFF	1A00-1BFF	Upper case and graphics characters
	35840-36351	8C00-8DFF	56320-56831	DC00-DDFF	1C00-1DFF	Reversed lower case characters
	36352-36863	8E00-8FFF	56832-57343	DE00-DEFF	1E00-1FFF	Reversed upper case and graphics

B Series Screen Map

1	D000	264832	493325	132525	3254	49252	56252	10252	15252	20252	25252	30252	35252	40252	45252	50252	55252	60252	65252	70252	75252	80252	85252	90252	95252	100252	105252	110252	115252	120252	125252	130252	135252	140252	145252	150252	155252	160252	165252	170252	175252	180252	185252	190252	195252	200252	205252	210252	215252	220252	225252	230252	235252	240252	245252	250252	255252	260252	265252	270252	275252	280252	285252	290252	295252	300252	305252	310252	315252	320252	325252	330252	335252	340252	345252	350252	355252	360252	365252	370252	375252	380252	385252	390252	395252	400252	405252	410252	415252	420252	425252	430252	435252	440252	445252	450252	455252	460252	465252	470252	475252	480252	485252	490252	495252	500252	505252	510252	515252	520252	525252	530252	535252	540252	545252	550252	555252	560252	565252	570252	575252	580252	585252	590252	595252	600252	605252	610252	615252	620252	625252	630252	635252	640252	645252	650252	655252	660252	665252	670252	675252	680252	685252	690252	695252	700252	705252	710252	715252	720252	725252	730252	735252	740252	745252	750252	755252	760252	765252	770252	775252	780252	785252	790252	795252	800252	805252	810252	815252	820252	825252	830252	835252	840252	845252	850252	855252	860252	865252	870252	875252	880252	885252	890252	895252	900252	905252	910252	915252	920252	925252	930252	935252	940252	945252	950252	955252	960252	965252	970252	975252	980252	985252	990252	995252	1000252																																																																																																												
2	D050	332832	328333	333334	338335	343336	348337	353338	358339	363340	368341	373342	378343	383344	388345	393346	398347	403348	408349	413350	418351	423352	428353	433354	438355	443356	448357	453358	458359	463360	468361	473362	478363	483364	488365	493366	498367	503368	508369	513370	518371	523372	528373	533374	538375	543376	548377	553378	558379	563380	568381	573382	578383	583384	588385	593386	598387	603388	608389	613390	618391	623392	628393	633394	638395	643396	648397	653398	658399	663400	668401	673402	678403	683404	688405	693406	698407	703408	708409	713410	718411	723412	728413	733414	738415	743416	748417	753418	758419	763420	768421	773422	778423	783424	788425	793426	798427	803428	808429	813430	818431	823432	828433	833434	838435	843436	848437	853438	858439	863440	868441	873442	878443	883444	888445	893446	898447	903448	908449	913450	918451	923452	928453	933454	938455	943456	948457	953458	958459	963460	968461	973462	978463	983464	988465	993466	998467	1000468																																																																																																																																																																																
3	D0A0	360840	3631	3654	3677	3699	3721	3743	3765	3787	3809	3831	3853	3875	3897	3919	3941	3963	3985	4007	4029	4051	4073	4095	4117	4139	4161	4183	4205	4227	4249	4271	4293	4315	4337	4359	4381	4403	4425	4447	4469	4491	4513	4535	4557	4579	4601	4623	4645	4667	4689	4711	4733	4755	4777	4799	4821	4843	4865	4887	4909	4931	4953	4975	4997	5019	5041	5063	5085	5107	5129	5151	5173	5195	5217	5239	5261	5283	5305	5327	5349	5371	5393	5415	5437	5459	5481	5503	5525	5547	5569	5591	5613	5635	5657	5679	5701	5723	5745	5767	5789	5811	5833	5855	5877	5899	5921	5943	5965	5987	6009	6031	6053	6075	6097	6119	6141	6163	6185	6207	6229	6251	6273	6295	6317	6339	6361	6383	6405	6427	6449	6471	6493	6515	6537	6559	6581	6603	6625	6647	6669	6691	6713	6735	6757	6779	6801	6823	6845	6867	6889	6911	6933	6955	6977	6999	7021	7043	7065	7087	7109	7131	7153	7175	7197	7219	7241	7263	7285	7307	7329	7351	7373	7395	7417	7439	7461	7483	7505	7527	7549	7571	7593	7615	7637	7659	7681	7703	7725	7747	7769	7791	7813	7835	7857	7879	7901	7923	7945	7967	7989	8011	8033	8055	8077	8099	8121	8143	8165	8187	8209	8231	8253	8275	8297	8319	8341	8363	8385	8407	8429	8451	8473	8495	8517	8539	8561	8583	8605	8627	8649	8671	8693	8715	8737	8759	8781	8803	8825	8847	8869	8891	8913	8935	8957	8979	9001	9023	9045	9067	9089	9111	9133	9155	9177	9199	9221	9243	9265	9287	9309	9331	9353	9375	9397	9419	9441	9463	9485	9507	9529	9551	9573	9595	9617	9639	9661	9683	9705	9727	9749	9771	9793	9815	9837	9859	9881	9903	9925	9947	9969	9991	10001																					
4	D0F0	346848	4689	4690	4691	4692	4693	4694	4695	4696	4697	4698	4699	4700	4701	4702	4703	4704	4705	4706	4707	4708	4709	4710	4711	4712	4713	4714	4715	4716	4717	4718	4719	4720	4721	4722	4723	4724	4725	4726	4727	4728	4729	4730	4731	4732	4733	4734	4735	4736	4737	4738	4739	4740	4741	4742	4743	4744	4745	4746	4747	4748	4749	4750	4751	4752	4753	4754	4755	4756	4757	4758	4759	4760	4761	4762	4763	4764	4765	4766	4767	4768	4769	4770	4771	4772	4773	4774	4775	4776	4777	4778	4779	4780	4781	4782	4783	4784	4785	4786	4787	4788	4789	4790	4791	4792	4793	4794	4795	4796	4797	4798	4799	4800	4801	4802	4803	4804	4805	4806	4807	4808	4809	4810	4811	4812	4813	4814	4815	4816	4817	4818	4819	4820	4821	4822	4823	4824	4825	4826	4827	4828	4829	4830	4831	4832	4833	4834	4835	4836	4837	4838	4839	4840	4841	4842	4843	4844	4845	4846	4847	4848	4849	4850	4851	4852	4853	4854	4855	4856	4857	4858	4859	4860	4861	4862	4863	4864	4865	4866	4867	4868	4869	4870	4871	4872	4873	4874	4875	4876	4877	4878	4879	4880	4881	4882	4883	4884	4885	4886	4887	4888	4889	4890	4891	4892	4893	4894	4895	4896	4897	4898	4899	4900	4901	4902	4903	4904	4905	4906	4907	4908	4909	4910	4911	4912	4913	4914	4915	4916	4917	4918	4919	4920	4921	4922	4923	4924	4925	4926	4927	4928	4929	4930	4931	4932	4933	4934	4935	4936	4937	4938	4939	4940	4941	4942	4943	4944	4945	4946	4947	4948	4949	4950	4951	4952	4953	4954	4955	4956	4957	4958	4959	4960	4961	4962	4963	4964	4965	4966	4967	4968	4969	4970	4971	4972	4973	4974	4975	4976	4977	4978	4979	4980	4981	4982	4983	4984	4985	4986	4987	4988	4989	4990	4991	4992	4993	4994	4995	4996	4997	4998	4999	5000
5	D140	356658	587851	587852	587853	587854	587855	587856	587857	587858	587859	587860	587861	587862	587863	587864	587865	587866	587867	587868	587869	587870	587871	587872	587873	587874	587875	587876	587877	587878	587879	587880	587881	587882	587883	587884	587885	587886	587887	587888	587889	587890	587891	587892	587893	587894	587895	587896	587897	587898	587899	587900	587901	587902	587903	587904	587905	587906	587907	587908	587909	587910	587911	587912	587913	587914	587915	587916	587917	587918	587919	587920	587921	587922	587923	587924	587925	587926	587927	587928	587929	587930	587931	587932	587933	587934	587935	587936	587937	587938	587939	587940	587941	587942	587943	587944	587945	587946	587947	587948	587949	587950	587951	587952	587953	587954	587955	587956	587957	587958	587959	587960	587961	587962	587963	587964	587965	587966	587967	587968	587969	587970	587971	587972	587973	587974	587975	587976	587977	587978	587979	587980	587981	587982	587983	587984	587985	587986	587987	587988	587989	587990	587991	587992	587993	587994	587995	587996	587997	587998	587999	588000																																																																																																																																																																		
6	D190	364856	49856	4986	4987	4988	4989	4990	4991	4992	4993	4994	4995	4996	4997	4998	4999	5000	5001	5002	5003	5004	5005</																																																																																																																																																																																																																																																																																																			

POKE 36879, X :		Border							
Screen	BLK	WHT	RED	CYN	PUR	GRN	BLU	YEL	
BLK	8	9	10	11	12	13	14	15	
WHT	24	25	26	27	28	29	30	31	
RED	40	41	42	43	44	45	46	47	
CYN	56	57	58	59	60	61	62	63	
PUR	72	73	74	75	76	77	78	79	
GRN	88	89	90	91	92	93	94	95	
BLU	104	105	106	107	108	109	110	111	
YEL	120	121	122	123	124	125	126	127	
ORG	136	137	138	139	140	141	142	143	
Lt. ORG	152	153	154	155	156	157	158	159	
PNK	168	169	170	171	172	173	174	175	
Lt. CYN	184	185	186	187	188	189	190	191	
Lt. PUR	200	201	202	203	204	205	206	207	
Lt. GRN	216	217	218	219	220	221	222	223	
Lt. BLU	232	233	234	235	236	237	238	239	
Lt. YEL	248	249	250	251	252	253	254	255	

POKE 59520, R#	POKE 59521, Value
R0	Horizontal total number of characters on line (Nht) including horizontal retrace. (true value = number + 1)
R1	Horizontal number of characters displayed (Nhd)
R2	Distance (in characters) from left to right margin of screen + 1
R3	Sync width. Lo nybble is vertical sync width (in lines) Hi nybble is horizontal sync (in characters).
R4	Number of display lines including retrace (Nvt).
R5	Vertical position of the edge of the screen.
R6	Number of display lines on screen (Nvd)
R7	Height of upper edge from bottom of screen (in lines displayed)
R8	Interlace and Skew: - Bit 0 1 = interlaced mode 0 = non interlaced mode Bit 1 if Bit 0 = 1 then interlace and video mode Bit 2 not used Bit 3 not used Bit 4 1 = scan from 32770 in memory Bit 5 1 = scan from 32772 in memory Bit 6 cursor (not implemented on the PET) Bit 7 cursor (not implemented on the PET)
R9	Number of lines between top of one display line and top of the next
R10	Cursor (not implemented on the PET)
R11	Cursor (not implemented on the PET)
R12	Control Register: Bit 0 add 256 to start address (512 for 8032) Bit 1 add 512 to start address (1024 for 8032) Bit 2 invert flyback Bit 3 invert video signal Bit 4 use top half of 4K character generator Bit 5 (not implemented on the PET) Bit 6 (not implemented on the PET) Bit 7 not used
R13	Value + 32768 is address of first character (multiply by 2 for 8032)
R14	Cursor location HI (not implemented on the PET)
R15	Cursor location LO (not implemented on the PET)
R16	Light pen position HI (read only)
R17	Light pen position LO (read only)

Mde:	IMM	ZPg	Z,X	(I),X	(I),Y	ABS	A,X	A,Y
Byts:	2	2	2	2	2	3	3	3
ORA	09	05	15	01	11	0D	1D	19
AND	29	25	35	21	31	2D	3D	39
EOR	49	45	55	41	51	4D	5D	59
ADC	69	65	75	61	71	6D	7D	79
STA		85	95	81	91	8D	9D	99
LDA	A9	A5	B5	A1	B1	AD	BD	B9
CMP	C9	C5	D5	C1	D1	CD	DD	D9
SBC	E9	E5	F5	E1	F1	ED	FD	F9

Op Code ends in -1, -5, -9, or -D

Mde:	IMM	ZPg	Z,X	ABS	A,X
Byts:	2	2	2	3	3
BIT			24		2C
STY			84	94	8C
LDY	A0	A4	B4	AC	BC
CPY	C0	C4		CC	
CPX	E0	E4		EC	

Op Code ends in -0, -4, or -C

Jim Butterfield

Mde:	IMM	ZPg	Z,X	Z,Y	ABS	A,X	A,Y
Byts:	2	2	2	2	3	3	3
ASL		06	16		0E	1E	
ROL		26	36		2E	3E	
LSR		46	56		4E	5E	
ROR		66	76		6E	7E	
STX		86		96	8E		
LDX	A2	A6		B6	AE		BE
DEC		C6	D6		CE	DE	
INC		E6	F6		EE	FE	

Op Code ends in -2, -6, or -E

Branches -0		
BPL	10	BMI 30
BVC	50	BVS 70
BCC	90	BCS 80
BNE	D0	BEQ F0

Jumps		
Mde:	ABS	(IND)
JSR	20	
JMP	4C	6C

Single Byte Op Codes (* Accumulator Mode)																
	0-	1-	2-	3-	4-	5-	6-	7-	8-	9-	A-	B-	C-	D-	E-	F-
-0	BRK				RTI		RTS									
-8	PHP	CLC	PLP	SEC	PHA	CLI	PLA	SEI	DEY	TYA	TAY	CLV	INY	CLD	INX	SED
-A	ASL*		ROL*		LSR*		ROR*		TXA	TXS	TAX	TSX	DEX		NOP	

Hexadecimal Conversion Chart

Hex	-0	>1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F	-00	-000
0-	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	0
1-	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	256	4096
2-	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	512	8192
3-	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	768	12288
4-	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	1024	16384
5-	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	1280	20480
6-	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	1536	24576
7-	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	1792	28672
8-	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	2048	32768
9-	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	2304	36864
A-	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	2560	40960
B-	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	2816	45056
C-	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	3072	49152
D-	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	3328	53248
E-	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	3584	57344
F-	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	3840	61440

Bit Values

Bit	Dec	Hex
0	1	\$0001
1	2	0002
2	4	0004
3	8	0008
4	16	0010
5	32	0020
6	64	0040
7	128	0080
8	256	0100
9	512	0200
10	1024	0400
11	2048	0800
12	4096	1000
13	8192	2000
14	16384	4000
15	32768	8000

The value under Hi is POKEd into the Hi byte of the frequency registers (54273, 54280, 54287). Likewise with Lo (54272, 54279, 54286)

Note	Octave 0			Octave 1			Octave 2			Octave 3		
	Oscillator Frequency			Oscillator Frequency			Oscillator Frequency			Oscillator Frequency		
	Decimal =	Hi (x256) + Lo		Decimal =	Hi (x256) + Lo		Decimal =	Hi (x256) + Lo		Decimal =	Hi (x256) + Lo	
C	268	1	12	536	2	24	1072	4	48	2145	8	97
C#	284	1	28	568	2	56	1136	4	112	2273	8	225
D	301	1	45	602	2	90	1204	4	180	2408	9	104
D#	318	1	62	637	2	125	1275	4	251	2551	9	247
E	337	1	81	675	2	163	1351	5	71	2703	10	143
F	358	1	102	716	2	204	1432	5	152	2864	11	48
F#	379	1	123	758	2	246	1517	5	237	3034	11	218
G	401	1	145	803	3	35	1607	6	71	3215	12	143
G#	425	1	169	851	3	83	1703	6	167	3406	13	78
A	451	1	195	902	3	134	1804	7	12	3608	14	24
A#	477	1	221	955	3	187	1911	7	119	3823	14	239
B	506	1	250	1012	3	244	2025	7	233	4050	15	210

Note	Octave 4			Octave 5			Octave 6			Octave 7		
	Oscillator Frequency			Oscillator Frequency			Oscillator Frequency			Oscillator Frequency		
	Decimal =	Hi (x256) + Lo		Decimal =	Hi (x256) + Lo		Decimal =	Hi (x256) + Lo		Decimal =	Hi (x256) + Lo	
C	4291	16	195	8583	33	135	17167	67	15	34334	134	30
C#	4547	17	195	9094	35	134	18188	71	12	36376	142	24
D	4817	18	209	9634	37	162	19269	75	69	38539	150	139
D#	5103	19	239	10207	39	223	20415	79	191	40830	159	126
E	5407	21	31	10814	42	62	21629	84	125	43258	168	250
F	5728	22	96	11457	44	193	22915	89	131	45830	179	6
F#	6069	23	181	12139	47	107	24278	94	214	48556	189	172
G	6430	25	30	12860	50	60	25721	100	121	51443	200	243
G#	6812	26	156	13625	53	57	27251	106	115	54502	212	230
A	7217	28	49	14435	56	99	28871	112	199	57743	225	143
A#	7647	29	223	15294	59	190	30588	119	124	61176	238	248
B	8101	31	165	16203	63	75	32407	126	151	64814	253	46

CB2 Note Values

Values are for simple CB2 sound.

PET/CBM : POKE 59467,16 : POKE 59466, (Oct) : POKE 59464, X
 VIC 20 : POKE 37147,16 : POKE 37146, (Oct) : POKE 37144, X
 C64 : POKE 56587,16 : POKE 56586, (Oct) : POKE 56584, X

Note	Oct = 15		Oct = 51		Oct = 85	
	Octave 0	Octave 1	Octave 2	Octave 3	Octave 4	Octave 5
	B	251 ^b	125	251	125	251
C	238	118	238	118	238	118
C#	224	110	224	110	224	110
D	210	104	210	104	210	104
D#	199	99	199	99	199	99
E	188	93	188	93	188	93
F	177	88	177	88	177	88
F#	168	83	168	83	168	83
G	158	78	158	78	158	78
G#	149	74	149	74	149	74
A	140	69	140	69	140	69
A#	133	65	133	65	133	65

Based on formula: Note n = (Note n-1) / 2 ↑ (1/12)
 Reset Port with first POKE (above) ,0

Commodore 64 SID Envelope Rates

Master Volume (MV) = Lo nybble of 54296. MV and A/D/S/R Registers (R1 & R2) are write only.
 Voice1: 54277/8 • Voice2: 54284/5 • Voice3: 54291/2. See Memory Map.

Value Hi nybble Lo nybble		POKE R1, (Hi + Lo)		POKE R2, (Hi + Lo)	
		Hi nybble	Lo nybble	Hi nybble	Lo nybble
		Attack Rate 0 to peak	Decay Rate peak to SL	Sustain Level ^{val} 1/15th's of MV	Release rate SL to 0
0	0	2 ms	6 ms	0/15MV	6 ms
16	1	8 ms	24 ms	1/15MV	24 ms
32	2	16 ms	48 ms	2/15MV	48 ms
48	3	24 ms	72 ms	3/15MV	72 ms
64	4	38 ms	114 ms	4/15MV	114 ms
80	5	56 ms	168 ms	5/15MV	168 ms
96	6	68 ms	204 ms	6/15MV	204 ms
112	7	80 ms	240 ms	7/15MV	240 ms
128	8	100 ms	300 ms	8/15MV	300 ms
144	9	250 ms	750 ms	9/15MV	750 ms
160	10	500 ms	1.5 s	10/15MV	1.5 s
176	11	800 ms	2.4 s	11/15MV	2.4 s
192	12	1.0 s	3.0 s	12/15MV	3.0 s
208	13	3.0 s	9.0 s	13/15MV	9.0 s
224	14	5.0 s	15.0 s	14/15MV	15.0 s
240	15	8.0 s	24.0 s	= MV	24.0 s

VIC 20 Note Values

Where two values are shown, it is necessary to alternate between them to get the true note.
 Voice frequency registers are 36874/5/6. Noise reg is 36877. Volume is Lo nybble of 36878. See Memory Map

Note	Octave 0		Octave 1		Octave 2		Octave 3	
	Value	Mod. Val.	Value	Mod. Val.	Value	Mod. Val.	Value	Mod. Val.
C	131		192	195	224		239	240
C#	140		197		226		240	241
D	145		200		227	228		
D#	151		203		229			
E	158		206	207	231			
F	161	162	208	209	232			
F#	166	167	211	212	233			
G	173	174	214		234	235		
G#	178		216		238	236		
A	181	182	218	219	237			
A#	185	186	220	221	237	238		
B	189	190	222	223	239			

SuperChart: VIC 20 / Commodore 64

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
0	00		@	end-line	BRK	0
1	01		A		ORA(I,X)	1
2	02		B			2
3	03	stop	C			3
4	04		D			4
5	05	white	E		ORA Z	5
6	06		F		ASL Z	6
7	07		G			7
8	08	lock	H		PHP	8
9	09	unlock	I		ORA #	9
10	0A		J		ASL A	10
11	0B		K			11
12	0C		L			12
13	0D	car ret	M		ORA	13
14	0E	text	N		ASL	14
15	0F		O			15
16	10		P		BPL	16
17	11	cur down	Q		ORA(I),Y	17
18	12	reverse	R			18
19	13	cur home	S			19
20	14	delete	T			20
21	15		U		ORA Z,X	21
22	16		V		ASL Z,X	22
23	17		W			23
24	18		X		CLC	24
25	19		Y		ORA Y	25
26	1A		Z			26
27	1B		[27
28	1C	red	\			28
29	1D	cur right]		ORA X	29
30	1E	green	↑		ASL X	30
31	1F	blue	←			31
32	20	space	space	space	JSR	32
33	21	!	!	!	AND(I,X)	33
34	22	"	"	"		34
35	23	#	#	#		35
36	24	\$	\$	\$	BIT Z	36
37	25	%	%	%	AND Z	37
38	26	&	&	&	ROL Z	38
39	27	/	/	/		39
40	28	(((PLP	40
41	29)))	AND #	41
42	2A	*	*	*	ROL A	42
43	2B	+	+	+		43
44	2C	,	,	,	BIT	44
45	2D	-	-	-	AND	45
46	2E	.	.	.	ROL	46
47	2F	/	/	/		47
48	30	0	0	0	BMI	48
49	31	1	1	1	AND(I),Y	49
50	32	2	2	2		50
51	33	3	3	3		51
52	34	4	4	4		52
53	35	5	5	5	AND Z,X	53
54	36	6	6	6	ROL Z,X	54
55	37	7	7	7		55
56	38	8	8	8	SEC	56
57	39	9	9	9	AND Y	57
58	3A	:	:	:		58
59	3B	;	;	;		59
60	3C	<	<	<		60
61	3D	=	=	=	AND X	61
62	3E	>	>	>	ROL X	62
63	3F	?	?	?		63

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
64	40	@	␣	@	RTI	64
65	41	A	␣,a	A	EOR(I,X)	65
66	42	B	␣,b	B		66
67	43	C	␣,c	C		67
68	44	D	␣,d	D		68
69	45	E	␣,e	E	EOR Z	69
70	46	F	␣,f	F	LSR Z	70
71	47	G	␣,g	G		71
72	48	H	␣,h	H	PHA	72
73	49	I	␣,i	I	EOR #	73
74	4A	J	␣,j	J	LSR A	74
75	4B	K	␣,k	K		75
76	4C	L	␣,l	L	JMP	76
77	4D	M	␣,m	M	EOR	77
78	4E	N	␣,n	N	LSR	78
79	4F	O	␣,o	O		79
80	50	P	␣,p	P	BVC	80
81	51	Q	␣,q	Q	EOR(I),Y	81
82	52	R	␣,r	R		82
83	53	S	␣,s	S		83
84	54	T	␣,t	T		84
85	55	U	␣,u	U	EOR Z,X	85
86	56	V	␣,v	V	LSR Z,X	86
87	57	W	␣,w	W		87
88	58	X	␣,x	X	CLI	88
89	59	Y	␣,y	Y	EOR Y	89
90	5A	Z	␣,z	Z		90
91	5B	[␣	[91
92	5C	£	␣	£		92
93	5D]	␣]	EOR X	93
94	5E	↑	␣	↑	LSR X	94
95	5F	←	␣	←		95
96	60		␣		RTS	96
97	61		␣		ADC(I,X)	97
98	62		␣			98
99	63		␣			99
100	64		␣			100
101	65		␣		ADC Z	101
102	66		␣		ROR Z	102
103	67		␣			103
104	68		␣		PLA	104
105	69		␣		ADC #	105
106	6A		␣		ROR A	106
107	6B		␣			107
108	6C		␣		JMP(I)	108
109	6D		␣		ADC	109
110	6E		␣		ROR	110
111	6F		␣			111
112	70		␣		BVS	112
113	71		␣		ADC(I),Y	113
114	72		␣			114
115	73		␣			115
116	74		␣			116
117	75		␣		ADC Z,X	117
118	76		␣		ROR Z,X	118
119	77		␣			119
120	78		␣		SEI	120
121	79		␣		ADC Y	121
122	7A		␣			122
123	7B		␣			123
124	7C		␣			124
125	7D		␣		ADC X	125
126	7E		␣		ROR X	126
127	7F		␣			127

Jim Butterfield

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
128	80		@	END		128
129	81	orange	A	FOR	STA(I,X)	129
130	82		B	NEXT		130
131	83	load & run	C	DATA		131
132	84		D	INPUT#	STY Z	132
133	85	f1	E	INPUT	STA Z	133
134	86	f2	F	DIM	STX Z	134
135	87	f3	G	READ		135
136	88	f4	H	LET	DEY	136
137	89	f5	I	GOTO		137
138	8A	f6	J	RUN	TXA	138
139	8B	f7	K	IF		139
140	8C	f8	L	RESTORE	STY	140
141	8D	car ret	M	GOSUB	STA	141
142	8E	graphics	N	RETURN	STX	142
143	8F		O	REM		143
144	90	black	P	STOP	BCC	144
145	91	cur up	Q	ON	STA(I),Y	145
146	92	rvs off	R	WAIT		146
147	93	clear	S	LOAD		147
148	94	insert	T	SAVE	STY Z,X	148
149	95	brown	U	VERIFY	STA Z,X	149
150	96	lt. red	V	DEF	STX Z,Y	150
151	97	dk. grey	W	POKE		151
152	98	md. grey	X	PRINT#	TYA	152
153	99	lt. green	Y	PRINT	STA Y	153
154	9A	lt. blue	Z	CONT	TXS	154
155	9B	lt. grey	[LIST		155
156	9C	magenta	{	CLR		156
157	9D	cur left		CMD	STA X	157
158	9E	yellow	↑	SYS		158
159	9F	cyan	←	OPEN		159
160	A0	□	!	CLOSE	LDY #	160
161	A1	▣	!"	GET	LDA(I,X)	161
162	A2	▤	,"	NEW	LDX #	162
163	A3	▥	#	TAB(163
164	A4	▦	\$	TO	LDY Z	164
165	A5	▧	%	FN	LDA Z	165
166	A6	▨	&	SPC(LDX Z	166
167	A7	▩	/	THEN		167
168	A8	▪	(NOT	TAY	168
169	A9	▫)	STEP	LDA #	169
170	AA	▬	*	+	TAX	170
171	AB	▭	+	-		171
172	AC	▮	,	*	LDY	172
173	AD	▯	-	/	LDA	173
174	AE	▰	.	↑	LDX	174
175	AF	▱	/	AND		175
176	B0	▲	0	OR	BCS	176
177	B1	△	1	>	LDA(I),Y	177
178	B2	▴	2	=		178
179	B3	▵	3	<		179
180	B4	▶	4	SGN	LDY Z,X	180
181	B5	▷	5	INT	LDA Z,X	181
182	B6	▸	6	ABS	LDX Z,Y	182
183	B7	▹	7	USR		183
184	B8	►	8	FRE	CLV	184
185	B9	▻	9	POS	LDA Y	185
186	BA	▼	:	SQR	TSX	186
187	BB	▽	;	RND		187
188	BC	▾	<	LOG	LDY X	188
189	BD	▿	=	EXP	LDA X	189
190	BE	▸	>	COS	LDX Y	190
191	BF	▹	?	SIN		191

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
192	C0	▩	▩	TAN	CPY #	192
193	C1	▫,a	▫	ATN	CMP(I),X	193
194	C2	▬,b	▬	PEEK		194
195	C3	▭,c	▭	LEN		195
196	C4	▮,d	▮	STR\$	CPY Z	196
197	C5	▯,e	▯	VAL	CMP Z	197
198	C6	▰,f	▰	ASC	DEC Z	198
199	C7	▱,g	▱	CHR\$		199
200	C8	▲,h	▲	LEFT\$	INY	200
201	C9	△,i	△	RIGHT\$	CMP #	201
202	CA	▴,j	▴	MID\$	DEX	202
203	CB	▵,k	▵	GO		203
204	CC	▶,l	▶		CPY	204
205	CD	▷,m	▷		CMP	205
206	CE	▸,n	▸		DEC	206
207	CF	▹,o	▹			207
208	D0	►,p	►		BNE	208
209	D1	▻,q	▻		CMP(I),Y	209
210	D2	▼,r	▼			210
211	D3	▽,s	▽			211
212	D4	▾,t	▾			212
213	D5	▿,u	▿		CMP Z,X	213
214	D6	▸,v	▸		DEC Z,X	214
215	D7	▹,w	▹			215
216	D8	►,x	►		CLD	216
217	D9	▻,y	▻		CMP Y	217
218	DA	▼,z	▼			218
219	DB	▽	▽			219
220	DC	▾	▾			220
221	DD	▿	▿		CMP X	221
222	DE	▸	▸		DEC X	222
223	DF	▹	▹			223
224	E0	►	►		CPX #	224
225	E1	▻	▻		SBC(I),X	225
226	E2	▼	▼			226
227	E3	▽	▽			227
228	E4	▾	▾		CPX Z	228
229	E5	▿	▿		SBC Z	229
230	E6	▸	▸		INC Z	230
231	E7	▹	▹			231
232	E8	►	►		INX	232
233	E9	▻	▻		SBC #	233
234	EA	▼	▼		NOP	234
235	EB	▽	▽			235
236	EC	▾	▾		CPX	236
237	ED	▿	▿		SBC	237
238	EE	▸	▸		INC	238
239	EF	▹	▹			239
240	F0	►	►		BEQ	240
241	F1	▻	▻		SBC(I),Y	241
242	F2	▼	▼			242
243	F3	▽	▽			243
244	F4	▾	▾			244
245	F5	▿	▿		SBC Z,X	245
246	F6	▸	▸		INC Z,X	246
247	F7	▹	▹			247
248	F8	►	►		SED	248
249	F9	▻	▻		SBC Y	249
250	FA	▼	▼			250
251	FB	▽	▽			251
252	FC	▾	▾			252
253	FD	▿	▿		SBC X	253
254	FE	▸	▸		INC X	254
255	FF	▹	▹		π	255

Reverse of ASCII

SuperChart: BASIC 2.0 / 4.0

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
0	00		@	end-line	BRK	0
1	01		A		ORA(I,X)	1
2	02		B			2
3	03	stop	C			3
4	04		D			4
5	05		E		ORA Z	5
6	06		F		ASL Z	6
7	07	bell	G			7
8	08		H		PHP	8
9	09	tab	I		ORA #	9
10	0A		J		ASL A	10
11	0B		K			11
12	0C		L			12
13	0D	car ret	M		ORA	13
14	0E	text	N		ASL	14
15	0F	top left	O			15
16	10		P		BPL	16
17	11	cur down	Q		ORA(I),Y	17
18	12	reverse	R			18
19	13	cur home	S			19
20	14	delete	T			20
21	15	del line	U		ORA Z,X	21
22	16	ers start	V		ASL Z,X	22
23	17		W			23
24	18		X		CLC	24
25	19	scroll dn	Y		ORA Y	25
26	1A		Z			26
27	1B	escape	[27
28	1C		\			28
29	1D	cur right]		ORA X	29
30	1E		↑		ASL X	30
31	1F		←			31
32	20	space	space	space	JSR	32
33	21	!	!	!	AND(I,X)	33
34	22	"	"	"		34
35	23	#	#	#		35
36	24	\$	\$	\$	BIT Z	36
37	25	%	%	%	AND Z	37
38	26	&	&	&	ROL Z	38
39	27	/	/	/		39
40	28	(((PLP	40
41	29)))	AND #	41
42	2A	*	*	*	ROL A	42
43	2B	+	+	+		43
44	2C	,	,	,	BIT	44
45	2D	-	-	-	AND	45
46	2E	.	.	.	ROL	46
47	2F	/	/	/		47
48	30	0	0	0	BMI	48
49	31	1	1	1	AND(I),Y	49
50	32	2	2	2		50
51	33	3	3	3		51
52	34	4	4	4		52
53	35	5	5	5	AND Z,X	53
54	36	6	6	6	ROL Z,X	54
55	37	7	7	7		55
56	38	8	8	8	SEC	56
57	39	9	9	9	AND Y	57
58	3A	:	:	:		58
59	3B	;	;	;		59
60	3C	<	<	<		60
61	3D	=	=	=	AND X	61
62	3E	>	>	>	ROL X	62
63	3F	?	?	?		63

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
64	40	@	␣	@	RTI	64
65	41	A	␣,a	A	EOR(I,X)	65
66	42	B	␣,b	B		66
67	43	C	␣,c	C		67
68	44	D	␣,d	D		68
69	45	E	␣,e	E	EOR Z	69
70	46	F	␣,f	F	LSR Z	70
71	47	G	␣,g	G		71
72	48	H	␣,h	H	PHA	72
73	49	I	␣,i	I	EOR #	73
74	4A	J	␣,j	J	LSR A	74
75	4B	K	␣,k	K		75
76	4C	L	␣,l	L	JMP	76
77	4D	M	␣,m	M	EOR	77
78	4E	N	␣,n	N	LSR	78
79	4F	O	␣,o	O		79
80	50	P	␣,p	P	BVC	80
81	51	Q	␣,q	Q	EOR(I),Y	81
82	52	R	␣,r	R		82
83	53	S	␣,s	S		83
84	54	T	␣,t	T		84
85	55	U	␣,u	U	EOR Z,X	85
86	56	V	␣,v	V	LSR Z,X	86
87	57	W	␣,w	W		87
88	58	X	␣,x	X	CLI	88
89	59	Y	␣,y	Y	EOR Y	89
90	5A	Z	␣,z	Z		90
91	5B	[␣	[91
92	5C	\	␣	\		92
93	5D]	␣]	EOR X	93
94	5E	↑	␣	↑	LSR X	94
95	5F	←	␣	←		95
96	60		␣		RTS	96
97	61		␣		ADC(I,X)	97
98	62		␣			98
99	63		␣			99
100	64		␣			100
101	65		␣		ADC Z	101
102	66		␣		ROR Z	102
103	67		␣			103
104	68		␣		PLA	104
105	69		␣		ADC #	105
106	6A		␣		ROR A	106
107	6B		␣			107
108	6C		␣		JMP(I)	108
109	6D		␣		ADC	109
110	6E		␣		ROR	110
111	6F		␣			111
112	70		␣		BVS	112
113	71		␣		ADC(I),Y	113
114	72		␣			114
115	73		␣			115
116	74		␣			116
117	75		␣		ADC Z,X	117
118	76		␣		ROR Z,X	118
119	77		␣			119
120	78		␣		SEI	120
121	79		␣		ADC Y	121
122	7A		␣			122
123	7B		␣			123
124	7C		␣			124
125	7D		␣		ADC X	125
126	7E		␣		ROR X	126
127	7F		␣			127

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL	
128	80		@	END		128	
129	81		A	FOR	STA(I,X)	129	
130	82		B	NEXT		130	
131	83	load & run	C	DATA		131	
132	84		D	INPUT#	STY Z	132	
133	85		E	INPUT	STA Z	133	
134	86		F	DIM	STX Z	134	
135	87	bell	G	READ		135	
136	88		H	LET	DEY	136	
137	89	set/clr tab	I	GOTO		137	
138	8A		J	RUN	TXA	138	
139	8B		K	IF		139	
140	8C		L	RESTORE	STY	140	
141	8D	car ret	M	GOSUB	STA	141	
142	8E	graphics	N	RETURN	STX	142	
143	8F	bot right	O	REM		143	
144	90		P	STOP	BCC	144	
145	91	cur up	Q	ON	STA(I),Y	145	
146	92	rvs off	R	WAIT		146	
147	93	clear	S	LOAD		147	
148	94	insert	T	SAVE	STY Z,X	148	
149	95	ins line	U	VERIFY	STA Z,X	149	
150	96	ers end	V	DEF	STX Z,Y	150	
151	97		W	POKE		151	
152	98		X	PRINT#	TYA	152	
153	99	scroll up	Y	PRINT	STA Y	153	
154	9A		Z	CONT	TXS	154	
155	9B	escape	[LIST		155	
156	9C		\	CLR		156	
157	9D	cur left	↑	CMD	STA X	157	
158	9E		↑	SYS		158	
159	9F		←	OPEN		159	
160	A0		█	CLOSE	LDY #	160	
161	A1		█	!	GET	LDA(I,X)	161
162	A2		█	"	NEW	LDX #	162
163	A3		█	#	TAB(163
164	A4		█	\$	TO	LDY Z	164
165	A5		█	%	FN	LDA Z	165
166	A6		█	&	SPC(LDX Z	166
167	A7		█	/	THEN		167
168	A8		█	(NOT	TAY	168
169	A9		█)	STEP	LDA #	169
170	AA		█	*	+	TAX	170
171	AB		█	+	-		171
172	AC		█	,	*	LDY	172
173	AD		█	-	/	LDA	173
174	AE		█	↑	↑	LDX	174
175	AF		█	/	AND		175
176	B0		█	0	OR	BCS	176
177	B1		█	1	>	LDA(I),Y	177
178	B2		█	2	=		178
179	B3		█	3	<		179
180	B4		█	4	SGN	LDY Z,X	180
181	B5		█	5	INT	LDA Z,X	181
182	B6		█	6	ABS	LDX Z,Y	182
183	B7		█	7	USR		183
184	B8		█	8	FRE	CLV	184
185	B9		█	9	POS	LDA Y	185
186	BA		█	:	SQR	TSX	186
187	BB		█	;	RND		187
188	BC		█	<	LOG	LDY X	188
189	BD		█	=	EXP	LDA X	189
190	BE		█	>	COS	LDX Y	190
191	BF		█	?	SIN		191

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
192	C0		▣	TAN	CPY #	192
193	C1	▣,a		ATN	CMP(I),X	193
194	C2	▣,b		PEEK		194
195	C3	▣,c		LEN		195
196	C4	▣,d		STR\$	CPY Z	196
197	C5	▣,e		VAL	CMP Z	197
198	C6	▣,f		ASC	DEC Z	198
199	C7	▣,g		CHR\$		199
200	C8	▣,h		LEFT\$	INY	200
201	C9	▣,i		RIGHT\$	CMP #	201
202	CA	▣,j		MID\$	DEX	202
203	CB	▣,k		GO		203
204	CC	▣,l		CONCAT	CPY	204
205	CD	▣,m		DOPEN	CMP	205
206	CE	▣,n		DCLOSE	DEC	206
207	CF	▣,o		RECORD		207
208	D0	▣,p		HEADER	BNE	208
209	D1	▣,q		COLLECT	CMP(I),Y	209
210	D2	▣,r		BACKUP		210
211	D3	▣,s		COPY		211
212	D4	▣,t		APPEND		212
213	D5	▣,u		DSAVE	CMP Z,X	213
214	D6	▣,v		DLOAD	DEC Z,X	214
215	D7	▣,w		CATALOG		215
216	D8	▣,x		RENAME	CLD	216
217	D9	▣,y		SCRATCH	CMP Y	217
218	DA	▣,z		DIRECTORY		218
219	DB	▣				219
220	DC	▣				220
221	DD	▣			CMP X	221
222	DE	▣			DEC X	222
223	DF	▣				223
224	E0	▣			CPX #	224
225	E1	▣			SBC(I),X	225
226	E2	▣				226
227	E3	▣				227
228	E4	▣			CPX Z	228
229	E5	▣			SBC Z	229
230	E6	▣			INC Z	230
231	E7	▣				231
232	E8	▣			INX	232
233	E9	▣			SBC #	233
234	EA	▣			NOP	234
235	EB	▣				235
236	EC	▣			CPX	236
237	ED	▣			SBC	237
238	EE	▣			INC	238
239	EF	▣				239
240	F0	▣			BEQ	240
241	F1	▣			SBC(I),Y	241
242	F2	▣				242
243	F3	▣				243
244	F4	▣				244
245	F5	▣			SBC Z,X	245
246	F6	▣			INC Z,X	246
247	F7	▣				247
248	F8	▣			SED	248
249	F9	▣			SBC Y	249
250	FA	▣				250
251	FB	▣				251
252	FC	▣				252
253	FD	▣			SBC X	253
254	FE	▣			INC X	254
255	FF	▣				255

Reverse of ASCII

BASIC 2.0 / BASIC 4.0 Memory Map

Reference to DOS, MLM, 80-Column, or those marked with an * are for BASIC 4.0 only. There are some differences between usage for the 40- and 80-column machines. 32K BASIC 4.0 Zero Page contents for 4000 and 8000 series machines after power up by Richard Evers. BASIC 2.0 Zero Page contents are mostly identical except for vectors.

Location		Contents				Description	Location		Contents				Description
Hex	Dec	4000 Hex Dec	8000 Hex Dec	4000 Hex Dec	8000 Hex Dec		Hex	Dec	4000 Hex Dec	8000 Hex Dec	4000 Hex Dec	8000 Hex Dec	
00-02	00-02	0 4C	76 4C	76		4C	76	FF 255	FF 255			USR jump instruction	
	01	1 73	115 73	115		4D	77	16 22	00 0				
	02	2 C3	195 C3	195		4E	78	00 0	FF 255			:JMP \$C373	
03	03	3 22	34 22	34		4F	79	00 0	00 0				
04	04	4 4	00 0	0		50	80	03 3	03 3			Search character	
	05	5 5B	91 5B	91		51	81	4C 76	4C 76				
	06	6 6	00 0	FF 255		52	82	43 67	FF 255			Scan-between-quotes flag	
	07	7 7	00 0	0		53	83	00 0	00 0				
	08	8 8	00 0	0		54	84	FF 255	FF 255			Input buffer pointer	
	09	9 9	04 4	04 4		55	85	87 135	00 0				
0A	0A	10 10	00 0	0		56	86	04 4	FF 255			Default DIM flag	
	0B	11 11	00 0	0		57	87	80 128	00 0				
	0C	12 12	00 0	FF 255		58	88	03 3	FF 255			Type: \$FF=string, \$00=numeric	
0D-0F	0D-15	13 00	00 0	0		59	89	00 0	00 0				
	0E	14 FF	255 FF	255		5A	90	00 0	00 0			Type: \$80=integer, 00=floating pt	
	0F	15 00	0 00	0		5B	91	00 0	00 0				
10	10	16 00	0 00	0		5C	92	00 0	00 0			Flag: DATA scan; LIST quote; memory	
	11	17 72	114 72	114		5D	93	00 0	00 0				
	12	18 D4	212 D4	212		5E	94	90 144	94 144			Subscript flag; FNx flag	
13-15	13-21	19 16	22 16	22		5F	95	00 0	00 0				
	14	20 13	19 13	19		60	96	00 0	00 0			0=INPUT; \$40=GET; \$98=READ	
	15	21 00	0 00	0		61	97	D4 212	D4 212				
	16	22 08	8 08	8		62	98	72 114	72 114			ATN sign/comparison evaluation flag	
	17	23 12	18 12	18		63	99	00 0	00 0				
	18	24 B3	179 B3	179		64	100	00 0	00 0			Disk status DS\$ descriptor	
	19	25 00	0 00	0		65	101	00 0	00 0				
	1A	26 FF	255 FF	255		66	102	90 144	90 144			Current I/O prompt flag	
	1B	27 00	0 00	0		67	103	D4 212	D4 212				
	1C	28 FF	255 FF	255		68	104	6C 108	6C 108			Integer value (for SYS, GOTO etc.)	
	1D	29 00	0 00	0		69	105	00 0	00 0				
	1E	30 FF	255 FF	255		6A	106	00 0	00 0			Pointers for descriptor stack	
1F-22	1F-34	31 40	64 40	64		6B	107	00 0	00 0				
	20	32 B2	178 B2	178		6C	108	00 0	00 0			Descriptor stack (temporary strings)	
	21	33 E9	233 E9	233		6D	109	00 0	00 0				
	22	34 CE	206 CE	206		6E	110	0A 0A	0A 0A			Utility pointer area	
	23	35 48	72 00	0		6F	111	B3 179	B3 179				
	24	36 00	0 FF	255		70	112	E6 230	E6 230			Product area for multiplication	
	25	37 00	0 00	0		71	113	77 119	77 119				
	26	38 00	0 FF	255		72	114	D0 208	D0 208			CHRGET subroutine; get BASIC char	
	27	39 00	0 00	0		73	115	02 2	02 2				
	28	40 01	1 01	1		74	116	E6 230	E6 230			:INC \$77	
	29	41 04	4 04	4		75	117	78 120	78 120				
2A-2B	2A-43	42 03	3 03	3		76	118	AD 173	AD 173			:BNE \$0076	
	2B	43 04	4 04	4		77	119	02 2	02 2				
2C-2D	2C-45	44 03	3 03	3		78	120	02 2	02 2			:INC \$78	
	2D	45 04	4 04	4		79	121	C9 201	C9 201				
2E-2F	2E-47	46 03	3 03	3		7A	122	3A 58	3A 58			:LDA \$0202	
	2F	47 04	4 04	4		7B	123	B0 176	B0 176				
30-31	30-49	48 00	0 00	0		7C	124	0A 10	0A 10			:CMP #\$3A	
	31	49 80	128 80	128		7D	125	C9 201	C9 201				
32-33	32-51	50 FE	254 FF	255		7E	126	20 32	20 32			:BCS \$0087	
	33	51 7F	127 00	0		7F	127	F0 240	F0 240				
34-35	34-53	52 00	0 00	0		80	128	EF 239	EF 239			:CMP #\$20	
	35	53 80	128 80	128		81	129	38 56	38 56				
36-37	36-55	54 14	20 FF	255		82	130	E9 233	E9 233			:BEQ \$0070	
	37	55 FF	255 FF	255		83	131	30 48	30 48				
38-39	38-57	56 00	0 FF	255		84	132	38 56	38 56			:SEC	
	39	57 80	128 00	0		85	133	E9 233	E9 233				
3A-3B	3A-59	58 01	1 FF	255		86	134	D0 208	D0 208			:SBC #\$30	
	3B	59 00	0 00	0		87	135	60 96	60 96				
3C-3D	3C-61	60 00	0 FF	255		77-78	77	119-120	119 02	2 02	2	:SBC #\$D0	
	3D	61 50	80 00	0		78	120	02 2	02 2				
3E-3F	3E-63	62 00	0 00	0		88-8C	88	136-140	136 80	128 80	128	:RTS	
	3F	63 04	4 04	4		89	137	4F 79	4F 79				
40-41	40-65	64 00	0 FF	255		8A	138	C7 199	C7 199			BASIC pointer (within subroutine)	
	41	65 00	0 00	0		8B	139	52 82	52 82				
42-43	42-67	66 04	4 FF	255		8C	140	F4 244	FF 255			Random number seed	
	43	67 00	0 00	0		8D-8F	8D	141-143	141 00	0 00	0		
44-45	44-69	68 24	36 24	36		8E	142	15 21	08 8			Jiffy clock for TI and TIS	
	45	69 04	4 00	0		8F	143	89 137	1F 31				
46-47	46-71	70 82	130 FF	255		90-91	90	144-145	144 55	85 55	85	Hardware interrupt vector IRQ	
	47	71 04	4 00	0		91	145	E4 228	E4 228				
48-49	48-73	72 FF	255 FF	255		92-93	92	146-147	146 78	120 78	120	BRK interrupt vector	
	49	73 00	0 00	0		93	147	D4 212	D4 212				
4A	4A	74 00	0 00	0		94-95	94	148-149	148 FF	255 FF	255	NMI interrupt vector	
4B-50	4B-80	75 00	0 00	0		95	149	B3 179	B3 179				

VIC 20 Standard Configuration

FFFF		65535
	8K Kernal ROM	
E000		57344
	8K BASIC ROM	
C000		49152
A000		40960
95FF	Colour Nybble Area	38399
9600	VIC Chip & I/O	38400
9000	Character Set	36864
8000		32768
2000	1/2K Screen RAM from basic VIC 20	4096
1E00		7680
	3 1/2 K RAM for BASIC	
1000		4096
0400		1024
0000	1K RAM Work Space	0

VIC 20 Expansion RAM Memory Changes

Exp RAM at:	BASIC Text	Screen	Colour Table
none	4096 / \$1000	7680 / \$1E00	38400 / \$9600
1024 / 4095*	1024 / \$0400	7680 / \$1E00	38400 / \$9600
8192 and up	4608 / \$1200	4096 / \$1000	37888 / \$9400

* VIC 1210 3K RAM Expander

VIC 20 With 40K RAM David Berezowski

- VIC 1020 Expansion Module Required with:
 1 - VIC 1210 3K RAM
 2 - VIC 1110 8K RAM (Switches 2,3,4 down - Switch 1 up)
 3 - VIC 1110 8K RAM (Switches 1,3,4 down - Switch 2 up)
 4 - VIC 1111 16K RAM

FFFF		65535
	8K Kernal ROM	
E000		57344
	8K BASIC ROM	
C000		49152
A000		40960
95FF	Colour Nybble Area	38399
9400	VIC Chip & I/O	37888
9000	Character Set	36864
8000		32768
	VIC 1110 8K RAM (2) (usable only with PEEK, POKE & M/L)	
	VIC 1111 16K RAM (4)	27 1/2 K for BASIC
	3 1/2 K of RAM from basic VIC 20	
1200		4608
1000	1/2K Screen RAM from basic VIC 20	4096
	VIC 1210 3K RAM (1) (usable only with PEEK, POKE & M/L)	
0400		1024
0000	1K RAM Work Space	0

9000	Interlace	Left Margin (=5)	36864	
9001		Top Margin (=25)	36865	
9002	Screen Ad Bit 9	Number of Columns (= 22)	36866	
9003	Bit 0	Number of Rows (= 23) Double Char	36867	
9004		Input Raster Value: Bits 1-8	36868	
9005	Screen Address Bits 13-10	Character Address Bits 13-10	36869	
9006	Light Pen Input	Horizontal	36870	
9007		Vertical	36871	
9008	Paddle Input	X	36872	
9009		Y	36873	
900A	ON	Voice 1 Frequency	36874	
900B	ON	Voice 2 Frequency	36875	
900C	ON	Voice 3 Frequency	36876	
900D	ON	Noise Frequency	36877	
900E	Multi Colour Mode	Sound Amplitude	36878	
900F	Background Colour	Foregnd/Backgnd	Border Colour	36879

6522 VIA 1

9110	DSR In	CTS In	DCD* In	RI* In	DTR Out	RTS Out	Data In	37136	
	RS-232 Interface or Parallel User Port								
9111	*Unused - see \$911F							37137	
9112	Data Direction Register B (for \$9110)							37138	
9113	Data Direction Register A (for \$911F)							37139	
9114	T1-L	RS 232 Send Speed;						37140	
9115	T1-H	Tape Write Timing						37141	
9116	T1-Latch L							37142	
9117	T1 Latch H							37143	
9118	T2-L	RS 232 Input Timing						37144	
9119	T2-H							37145	
911A	Shift Register (* unused)							37146	
911B	T1 Control	T2 Ctrl	Shift Register Control		PB LE	PA LE	37147		
911C	CB2: RS 232 Send		CB1 Ctrl	CA2: Tape Motor Ctrl		CA1 Ctrl	37148		
911D	NMI:	T1	T2	CB1: RS 232 In		CA1: RESTORE	37149		
911E	NMI En.	T1 Enab	T2 Enab	CB1 En.		CA1 En.	37150		
911F	ATN Out	Tape Sense	Fire	Joystick Left	Joystick Down	Joystick Up	Serial Data In	Serial Clock In	37151

6522 VIA 2

9120	Joystick Right	Tape Out	37152					
	Keyboard Row Select							
9121	Keyboard Column Input		37153					
9122	Data Direction Register B (for \$9120)		37154					
9123	Data Direction Register A (for \$9121)		37155					
9124	T1-L	Cassette Tape Read;	37156					
9125	T1-H	Keyboard and Clock	37157					
9126	T1-Latch L	Interrupt Timing	37158					
9127	T1 Latch H		37159					
9128	T2-L	Serial Bus Timing	37160					
9129	T2-H	Tape R/W Timing	37161					
912A	Shift Register (* unused)		37162					
912B	T1 Control	T2 Ctrl	Shift Register Control	PB LE	PA LE	37163		
912C	Serial Bus Data Out		CB1 Ctrl	Serial Clock Line Out		CA1 Ctrl	37164	
912D	IRQ:	T1	T2	CB1: SRQ In		CA1: Tape In	37165	
912E	IRQ En.	T1 Enab	T2 Enab	CB1 En.		CA1 En.	37166	
912F	*Unused (see \$9121)							37167



Table with 4 columns: Address, Description, Address, Description, Address, Description. Contains memory map details for Commodore 64, including registers like Chip I/O, Floating Point, and various control bits.

Commodore 64 ROM Routines

Table with 4 columns: Address, Description, Address, Description, Address, Description. Lists ROM routines such as ROM control vectors, Error messages, and various system functions.

6566 Video Chip

C64 Control & Miscellaneous Registers

D011	Extended Clr. Mode	Display Map	Bit Enable	Row Select	Y-Scroll	53265
D012	Raster Register					53266
D013	Light Pen Input					X 53267
D014						Y 53268
D016	x	x	Reset	Multi Colour	Column Select	X-Scroll 53270
D018	VM13	VM12	VM11	VM10	Character Base CB13 CB12 CB11	x 53272
D019	IRQ	Interrupt Sense:			Light Pen Spr-Spr Collision Spr-Back Collision	Raster 53273
D01A	Interrupt Enable:			Light Pen Spr-Spr Collisions Spr-Back Collisions	Raster 53274	
Colour Registers						
D020	X	Exterior Colour (Border)				53280
D021	X	Background Colour #0				53281
D022	X	Background Colour #1				53282
D023	X	Background Colour #2				53283
D024	X	Background Colour #3				53284
D025	X	Sprite MultiColour #0				53285
D026	X	Sprite MultiColour #1				53286

CIA 1 (IRQ) (6526)

\$DC00	Paddle Sel A B	Fire	Right	Joystick 0 Left	Down	Up	PRA	56320	
Keyboard Row Select (inverted)									
\$DC01	Fire	Right	Joystick 1 Left	Down	Up	PRB	56321		
Keyboard Column Read									
\$DC02	\$FF - All Output							DDRA	56322
\$DC03	\$00 - All Input							DDRB	56323
\$DC04	Timer A							TAL	56324
\$DC05	Timer B							TAH	56325
\$DC06	Timer A							TBL	56326
\$DC07	Timer B							TBH	56327
\$DC0D	Tape Input	One Shot	Out Mode	Timer PB6 Out	Interrupt A Start	ICR	56333		
\$DC0E	One Shot	Out Mode	Timer PB7 Out	Timer B Start	CRA	56334			
\$DC0F	One Shot	Out Mode	Timer PB7 Out	Timer B Start	CRB	56335			

CIA 2 (NMI) (6526)

\$DD00	Serial IN	Clock IN	Serial OUT	Clock OUT	ATN OUT	RS-232 OUT	VIC II addr 15	VIC II addr 14	PRA	56576		
\$DD01	DSR IN	CTS IN	DCD* IN	RI* IN	DTR OUT	RTS OUT	RS-232 IN	PRB	56577			
\$DD02	\$3F - Serial									DDRA	56578	
\$DD03	\$00 - P.U.P. All Input				or				\$06 - RS-232		DDRB	56579
\$DD04	Timer A									TAL	56580	
\$DD05	Timer B									TAH	56581	
\$DD06	Timer A									TBL	56582	
\$DD07	Timer B									TBH	56583	
\$DD0D	RS-232 IN	Timer B	Interrupt A	ICR	56589							
\$DD0E	Timer A Start	CRA	56590									
\$DD0F	Timer B Start	CRB	56591									

* Connected but not used by O.S.

C64 Sprite Registers

Sprite 0	Sprite 7	Sprite 0	Sprite 7
D000	D00E	X Position	53248 53262
D001	D00F	Y Position	53249 53263
Bit For Sprite#:			
7 6 5 4 3 2 1 0			
D010	X-Position High		53264
D015	Sprite Enable Flags		53269
D017	Y-Expand		53271
D01B	Background Priority		53275
D01C	Sprite MultiColour Mode		53276
D01D	X-Expand		53277
D01E	Interrupt: Sprite Collision		53278
D01F	Interrupt: Background Collision		53279

Processor I/O Port (6510)

\$0000	IN	IN	OUT	IN	OUT	OUT	OUT	OUT	DDR 0
\$0001	Tape Motor		Tape Sense	Tape Write	D-ROM Switch	EF RAM Switch	AB RAM Switch	PR 1	

SID (6581)

Voice 1	Voice 2	Voice 3	Frequency	L	Voice 1	Voice 2	Voice 3
\$D400	\$D407	\$D40E		H	54272	54279	54286
\$D401	\$D408	\$D40F		H	54273	54280	54287
\$D402	\$D409	\$D410	Pulse Width	L	54274	54281	54288
\$D403	\$D40A	\$D411	0 0 0 0	H	54275	54282	54289
\$D404	\$D40B	\$D412	Voice Type NSE PUL SAW TRI	Key	54276	54283	54290
\$D405	\$D40C	\$D413	Attack Time 2ms - 8ms	Decay Time 6ms - 24 sec	54277	54284	54291
\$D406	\$D40D	\$D414	Sustain Level	Release Time 6ms - 24 sec	54278	54285	54292

Voices (write only)

\$D415	0 0 0 0 0	L	54293
\$D416	Filter Frequency		H 54294
\$D417	Resonance	Filter Voices Ext V3 V2 V1	54295
\$D418	Passband V3 off HI BP LO	Master Volume	54296

Filter & Volume (write only)

\$D419	Paddle X (A/D *1)	54297
\$D41A	Paddle Y (A/D *2)	54298
\$D41B	Noise 3 (random)	54299
\$D41C	Envelope 3	54300

Sense (read only)

Note: Special Voice Features (TEST, RING MOD, SYNC) are omitted from the above diagram.

Commodore B128 ROM Routines

The following is a map of routines and data within the current (September 1983) version of the Commodore B128 computer. Caution: The same routines exist in the B256 but the addresses are not exactly the same.

8000	Jumps: Warm start, Cold start	8E24	Perform [DISPOSE]
8006	Mask: 'CBMS'	8E7A	Perform [PRINT*]
800B	Reference Vectors (unused)	8E80	Perform [CMD]
8027	Action vectors	8E9D	Perform [PRINT]
803B	Action (run etc) vectors	8F15	Perform [GET]
8043	Function vectors	8F4B	Perform [INPUT*]
80F1	Operation vectors	8F66	Perform [INPUT]
80FE	Keywords	8FA8	Prompt & input
82BF	Message vectors	8FEA	Perform [READ]
82E7	Messages	90E7	Perform [SYS]
8550	Print 'Out of memory'	910C	Perform [DIM]
8552	Error routine	9116	Perform [DEF]
85AE	Print line number	9146	Perform [POKE]
85C0	Warm start	9152	Perform [WAIT]
85F3	Handle new line	917F	Perform [KEY]
8644	Rechain lines	91BC	Perform [VERIFY]
8645	Receive input line	91C8	Perform [LOAD]
871F	Find BASIC line	921B	Perform [SAVE]
8751	Command dispatcher	9243	Perform [OPEN]
87DB	Peek stack for FOR/GOSUB	9297	Perform [CLOSE]
8815	Open text space	92A1	Perform [CATALOG]
8866	Stack too deep?	936D	Perform [DOPEN]
8889	Check string space	937E	Perform [APPEND]
8890	Check BASIC space	93A9	Perform [DCL] [CLOSE]
889F	Check array space	93C3	Perform [DSAVE]
88AB	'out of array space'	93CC	Perform [DLOAD]
88BF	Crunch tokens	93DE	Perform [BANK]
898D	Perform [LIST]	93EC	Perform [BSAVE]
8A29	Perform [NEW]	940E	Perform [BLOAD]
8A45	Perform [CLR]	9427	Perform [HEADER]
8A90	'USING' characters	9464	Perform [SCRATCH]
8A94	Perform [FOR]	949E	Perform [RECORD]
8B06	Perform [NEXT]	9510	Perform [DCL] [LEAR]
8B79	Perform [RESTORE]	9513	Perform [COLLECT]
8BA8	Perform [STOP]	952A	Perform [COPY]
8BAA	Perform [END]	9546	Perform [COL] [OST]
8BE9	Perform [CONT]	9552	Perform [RENAME]
8C07	Perform [RUN]	9560	Perform [BACKUP]
8C25	Perform [GOSUB]	9586	Patch area
8C42	Perform [IF]	95C1	Evaluate expression
8C77	Perform [REM/ELSE]	95CF	Recursive entry
8C7C	Perform [GO]	96CB	Value of PI in binary
8C84	Perform [GOTO]	96F8	Evaluate [NOT]
8C8B	Perform [RETURN]	9724	Eval within parens
8CDF	Perform [DATA]	979A	Go for disk status
8CED	Next statement	986B	Evaluate [OR]
8CF0	Next line	986E	Evaluate [AND]
8D16	Perform [TRAP]	98A8	Evaluate [COMPARE]
8D2B	Perform [ON]	992C	Get var name/loc
8D4E	Get fixed point number	99BF	Check alphabetic
8D8A	Perform [LET]	9AF5	Array ptr subtrn
8DC4	Perform [RESUME]	9B06	Float-fixed

8BA4	'bad subscript'	98A4	Evaluate [numeric]
98A7	'illegal quantity'	98D7	Evaluate [PRINT]
98F5	Evaluate [FRE]	9933	Evaluate [POS]
9D33	Fixed-float	9D39	Confirm not direct
9D4A	Check direct mode	9D57	Evaluate [PEEK*]
9E07	Evaluate [subtract]	9E30	Evaluate [add]
9E4D	Evaluate [error]	9F5E	Overflow error
9FCA	Evaluate [LOG]	A00B	Evaluate [multiplying]
A0D0	+ 10 floating	A0E9	Evaluate [division]
A148	Error: 'divide by zero'	A210	Evaluate [SGN]
A22F	Evaluate [ABS]	A2B1	Evaluate [INT]
A3B4	Print numeric	A3C3	Print canned message
A50D	+ 32768	A537	Evaluate [SQRT]
A541	Evaluate [power]	A57A	Evaluate [negate]
A5B3	Evaluate [EXP]	A659	Evaluate [RD]
A6A6	Evaluate [COS]	A6AD	Evaluate [SIN]
A6F8	Evaluate [TAN]	A791	Evaluate [ATN]
A7C0	Perform [PUDEF]	A7DB	Evaluate [STR\$]
A805	Set up string descriptors	A81F	Scan and set up string
A8AB	Build string into memory	A8E6	Discard unwanted string
A955	clean descriptor stack	AAD1	Evaluate [CHR\$]
AAEB	Evaluate [LEFT\$]	AB22	Evaluate [RIGHT\$]
AB42	Evaluate [MID\$]	AB8E	Evaluate [LEN]
AB9D	Evaluate [ASC]	ABAE	Evaluate [VAL]
AD53	Allocate dynamic string space	ADB5	Garbage collection
AF9D	Perform [DELETE]	AFD4	Get line range
B026	Perform <PRINT USING>	B4B8	Reset text pointer
B4E5	Evaluate integer	B501	Evaluate numeric
B504	Check numeric mode	B506	Check string mode
B52E	Print format character	B53A	Print character
B7CB	Disk command formats		



6526 CIA 1

DB00	Inter-Processor Data					56064	
DB01	X	IRQ Out	X	X	SEMAPH	Busy	56065
DB02	Data Direction Register For DB00					56066	
DB02	Data Direction Register For DB01					56067	
Unused							
DB0D		IP Flag					56077
DB0E	Unused					56078	
DB0F	Unused					56079	

6526 CIA 2

DC00	IEEE Data In/Out					56320	
DC01	User Port					56321	
DC02	Data Direction Register For DC00					56322	
DC02	Data Direction Register For DC01					56323	
Unused							
DC06	Timer B					L	56326
DC07						H	56327
DC08						1/10 Sec.	56328
DC09						Sec.	56329
DC0A						Min.	56330
DC0B						Hour	56331
DC0C	Unused						56332
DC0D						Alarm	56333
DC0E	Unused						56334
DC0F	TOD		Timer Force		Timer Start		56335

6551 ACIA

DD00	Data Register										
DD01	IRQ	DSR	DCD	Tx	Ready	Rx	OV	Error FR	PA		56577
DD02	XTRR Stop	# of Bits		Clk	Speed						56578
DD03	Parity		Echo		Tx	IRQ	Rx	DTR			56579

6545 CRT Controller

D800	D801	Typical Value
55296	55297	(Decimal)
0	Horizontal Total	108 or 126 or 127
1	Horizontal Char Displayed	80
2	Horizontal Sync Position	83 or 98 or 96
3	Sync Width	15 or 10
	V	H
4	Vertical Total	25 or 31 or 38
5	Vert Total Adjust	3 or 6 or 1
6	Vertical Displayed	25
7	Vert. Sync Position	25 or 28 or 30
8	Mode	0
9	Scan Lines	13 or 7
10	Cursor Start	96 (blink) or 0 or 6 (underline)
11	Cursor End	13 or 7
12	Display Address	H 0
13		L 0
14	Cursor Address	H Varies
15		L Varies
16	Light Pen In	H 0
17		L 0

Most Register are Write Only 14/15 are Read/Write
16/17 are Read Only
Registers 10, 14 and 15 change as the cursor moves

6581 SID

DA01	Voice 1 Frequency High				55809	
DA04		Saw Tooth	Ring Mod	Key	55812	
DA05	Attack				Decay	55813
DA06	Sustain				Release	55814
DA0F	Voice 3 Modulating Freq Hi				55823	
DA18					Volume	55832



VIC 20 Super Expander (1211A)

Memory Map

General Input/Output Routines

A000 - A001	Vector: RESET (\$A044)
A002 - A003	Vector: NMI (\$A077)
A004 - A008	ROM identification ('a0CBM')
A009 - A010	Table: function key numbers
A011 - A043	Table: initial function key definitions
A044 - A076	RESET routine
A077 - A08A	NMI routine
A08B - A0BE	Parse KEY (get parameters and check syntax)
A0BF - A131	Display all function key definitions
A110 - A11C	Print " " + chr\$(34) and an optional '+'
A11D - A131	Print " " + chr\$(13) and an optional '+'
A132 - A135	Table: ASCII string for output ('key' backwards)
A136 - A13F	Table: ASCII string for output (" " + chr\$(13) backwards)
A140 - A149	Table: ASCII string for output (" " + chr\$(34) backwards)
A14A - A17A	Delete current function key string (key number in .X)
A17B - A1B0	Insert string into function key definition area
A1B1 - A1BE	Locate function key definition (key # in .X, return index in .Y)
A1BF - A213	Table: new BASIC keywords in ASCII form
A214 - A237	Table: vectors corresponding to new BASIC tokens (\$CC to \$DD)
A238 - A2A1	Initialize kernel vectors, I/O, RAM
A2A2 - A2C1	Table: kernel vectors (L,H)
A2C2 - A2C7	Warm start routine
A2C8 - A317	Output a character to device 3 (char in .A)
A318 - A336	End music mode
A337 - A365	Interpret keyboard matrix input
A366 - A369	Table: keyboard matrix code for function keys
A36A - A371	Table: conversion pattern for function keys
A372 - A394	IRQ routine
A395 - A3A5	Input a char from any device (device number in \$99)
A3A6 - A3B3	Output a char to any device (char in .A, device num in \$9A)
A3B4 - A3F1	Input each char from keyboard buffer
A3B4 - A3E7	Handle 'RUN' key
A3E8 - A3F1	Handle 'RETURN' key
A3F2 - A3FC	Input from device 0
A3FD - A406	Print an error message in GRAPHIC 0 mode (error token in .A)
A407 - A4B9	Lexically analyse BASIC source line (translate to tokens)
A4BA - A503	Print BASIC tokens in ASCII form
A504 - A529	Start new BASIC statement
A515 - A523	Handle new tokens (\$CC to \$D6)
A52A - A58A	Get and evaluate an expression
A558 - A58A	Handle new function tokens (\$D7 to \$DD)
A58B - A596	Table: BASIC vectors for RAM
A597 - A5A4	Change BASIC vectors during RESET

Music Routines

A5A5 - A5D0	Save current sound table (address of table in .X,.Y)
A5D1 - A601	IRQ music driver
A602 - A625	Table: conversion for note index to frequency
A626 - A6E5	Interpret music mode characters (char in .A)
A629 - A643	Execute 'O' command, default 3
A644 - A65D	Execute 'T' command, default 0
A65E - A674	Execute 'S' command, default 4
A675 - A686	Execute 'V' command, default 7
A687 - A693	Execute 'R' command
A694 - A69B	Execute 'P' command
A69C - A6A7	Execute 'Q' command
A6A8 -	Play new note (note index in .Y)
A6AB - A6B3	Save new sound table when previous note finishes
A6B4 - A6B9	Common return routine
A6BA - A6CD	Play notes 'A' to 'G'
A6CE - A6DA	Execute 'W' command
A6DB - A6E5	Execute '\$' command
A6E6 - A6EC	Table: conversion for notes to note index
A6ED - A6EF	Table: conversion for octave to base note index
A6F0 - A6F9	Table: conversion for tempo to duration (jiffies)

Parsing New Command Routines

A6FA -	Look for and evaluate first 1-byte and two 2-byte parameters
A6FD -	Look for and evaluate two 2-byte parameters
A700 -	Look for and evaluate one 2-byte parameter
A714 - A71B	Save one 1-byte parameter (parameter in .A, index in .Y)
A71C -	Look for and evaluate two 1-byte parameters
A71F - A72B	Look for and evaluate one 1-byte parameter

A72C - A73F	Parse GRAPHIC (get parameters and check syntax)
A740 - A762	Parse CIRCLE
A763 - A7A4	Parse DRAW
A7A5 - A7BC	Parse POINT
A7BD -	Parse COLOR
A7C8 - A7CE	Go to execute commands after parsing
A7CF - A7D8	Parse REGION
A7D9 - A7DC	Parse SCNCLR
A7DD - A7E9	Parse SOUND
A7EA - A809	Parse CHAR
A80A - A810	Parse PAINT
A811 - A817	Parse RPOT
A818 - A81B	Parse RPEN
A81C - A81F	Parse RSND
A820 - A823	Parse RCOL
A824 - A827	Parse RGR
A828 - A842	Parse RDOT
A843 - A846	Parse RJOY
A847 - A84E	Look for first 1-byte parameter
A84F - A866	Indirect jump to execute new commands (pointer to parameter save area in .X,.Y, command index in .A)
A867 - A878	Table: vector to execute new commands (H)
A879 - A88A	Table: vector to execute new commands (L)
A88B - AA22	Execute GRAPHIC
AA8B - A94E	Handle GRAPHIC 1,2,3 if previous was 0
AA8D - A942	Transfer BASIC program to above \$2000 and execute CLR
AA93 - A94E	Make screen at \$1E00 and character set at \$1000
AA94F - A9A8	Handle GRAPHIC 4
AA967 - A9AB	Transfer BASIC program down to old location and execute CLR
AA9AC - A9B7	Handle GRAPHIC 0 if previous was 1,2,3
AA9B8 - AA22	Set up proper GRAPHIC screen
AA23 - AA28	Execute RGR
AA29 - AA6A	Execute COLOR
AA68 - AA84	Execute REGION
AA85 - AA8B	Execute RCOL
AA8C - AA6E	Execute RDOT
AAE7 - AAF1	Execute POINT
AAF2 - AB12	Execute SCNCLR
AB13 - AB22	Execute DRAW (C TO X,Y ...)
AB23 - AB34	Execute DRAW (C,X,Y TO X,Y ...)
AB35 - AB54	Execute SOUND
AB55 - AB69	Execute RSND
AB6A - AB76	Execute RPOT
AB77 - AB7D	Execute RPEN
AB7E -	Plot a single point from parameter save area
AB86 - ABE4	Plot a single point from beginning scaled X,Y coordinates
ABE5 -	Set up pointers to char and colour memory
ABFA - AC0A	Set up pointer to colour memory
AC0B -	Draw a line with a new starting coordinate
AC11 - AC92	Draw a line starting from previous coordinate (using a version of Bresenham's DDA algorithm)
AC93 - AD12	Execute CIRCLE using principal of digital differential analyser (DDA)
AD13 -	Convert starting angle to radians
AD19 - AD22	Divide FAC#1 by 16
AD23 -	Calculate new scaled X and Y coordinate on locus
AD39 - AD6B	Calculate unit offset times scaled radius
AD6C - ADDE	Execute PAINT
ADDF -	Check for possible new lower bound pivot coordinate
ADE8 - AE01	Save pivot coordinate
AE02 - AE0B	Check for possible new upper bound pivot coordinate
AE0C -	Check if able to PAINT a coordinate
AE0F - AE1E	Check if able to PAINT a coordinate (X,Y in .A,.Y)
AE1F - AE23	Move beginning scaled X,Y coordinate to .A,.Y
AE24 - AE3B	Check if coordinate has been already plotted on
AE3C - AE44	Move beginning scaled X coordinate to the right
AE45 - AE51	Move beginning scaled X coordinate 2 to the left
AE52 - AE56	Flag 'FORMULA TOO COMPLEX' error message
AE57 - AED9	Execute CHAR
AEDA - AF13	Execute RJOY
AF14 - AF33	Set up correct VIC chip screen registers
AF34 -	Save number of coordinates and colour register
AF39 - AF3E	Save colour register
AF3F - AF47	Copy beginning from ending scaled X,Y coordinate
AF48 - AF75	Scale X and Y coordinates
AF76 - AFBO	Scale X and Y coordinates to the range 0 to 159 (X = .A*coordinate*2/256) (number of columns or rows in .A)

AFB1 - AFBA	Table: vector to map Y coordinate to colour memory (L)
AFB5 - AFCE	Table: vector to map X coordinate to character memory (L)
AFCF - AFE2	Table: vector to map X coordinate to character memory (H)
AFE3 - AFE5	Table: bit set for colour memory
AFE6 - AFE6	(not used - contains \$00)
AFE7 - AFEE	Table: bit mask for highres mode
AFEF - AFF6	Table: bit mask for multicolour mode
AFF7 - AFFA	Table: bytes to plot in multicolour mode
AFFB - AFFE	Table: conversion factor for VIC chip screen registers
AFFF - AFFF	(not used - contains \$AA)
General RAM Area	
0024	Number of coordinates
0024	Flag: colour register mode (\$FF = multicolour, \$00 = highres)
0024 - 0025	Pointer: new start of variables / start of BASIC memory
0026	Temp area for building VIC chip registers / for building character byte / for saving start of BASIC (L)
Current Coordinates	
0062	Ending scaled X coordinate (0 to 159)
0063	Beginning scaled X coordinate (0 to 159)
0064	Scaled X difference (absolute value)
0065	Ending scale Y coordinate (0 to 159)
0066	Beginning scaled Y coordinate (0 to 159)
0067	Scaled Y difference (absolute value-1)
For Scaling Coordinates	
0069	Multiplicand-1
006A	16-bit product
006B - 006C	10-bit multiplier
For DRAW	
0067	Scaled X unit direction-1
006A	Scaled Y unit direction
006B - 006C	Number of scaled Y units left to plot before next scaled X unit (count up)
006D - 006E	Number of points left to plot (count up)
For PAINT	
0069	Index: pivot coordinates save area
For CHAR	
0069	Current row (0 to 19)
006A	Current column (0 to 19)
006B	Length of string
006C - 006D	Pointer: string location
General RAM Area	
009B	Index: beginning of current function key definition
009B - 009C	Pointer: current char set address / byte in char set / position in screen memory / destination of byte of BASIC program to transfer
009D	Index: end of function key definition area
009E	Current function key number / length of current function key string
009F	Length of current function key string (count down)
009E - 009F	Pointer: byte in colour memory
00AC - 00AD	Pointer: current byte (function key definition, tape, scrolling)
00C3	Flag: 0 = have transferred BASIC program to a new location
00C3 - 00C4	Pointer: kernal set up / current music table / parameter save area (\$003C)
00FB - 00FC	Pointer: top of BASIC memory (usually same as \$0284-\$0285)
For KEY	
028F - 0290	Vector: interpret keyboard input (\$A337)
02A1	Number of bytes taken by Super Expander in high memory (\$88)
02A2	Number of characters in function key definition
02A3	Index: current byte of function key string
02A4	Length of function key string (amount left to output)
For Music	
02A5	Previous character in music mode
02A6	Music mode (\$80 = on)
02A7	Screen echo (\$50 = on, \$00 = off)

02A8	Current voice (sound register-1)
02A9	Current note index
02AA	Current duration (jiffies)
02AB	Current sound amplitude (volume * 2)
02AC	Current octave (base note index)
02AD	Voice 1 note index (+ \$80)
02AE	Voice 1 duration count down (jiffies)
02AF	Voice 2 note index (+ \$80)
02B0	Voice 2 duration count down (jiffies)
02B1	Voice 3 note index (+ \$80)
02B2	Voice 3 duration count down (jiffies)
02B3	Voice 4 note index (+ \$80)
02B4	Voice 4 duration count down (jiffies)
02B5 - 02BF	(for expansion)
For Execution Of New Commands	
02C0 - 02C2	Jump table: execute new commands (JMP \$A84F)
02C3	Current VIC chip left margin register
02C4	Current VIC chip top margin register
02C5	Current VIC chip number of columns register
02C6	Current VIC chip number of rows register
02C7	Current row of cursor
02C8	Current GRAPHIC mode
02C9	(for expansion)
02CA	Current colour register parameter (while plotting)
02CB	Current screen colour
02CC	Current border colour
02CD	Current character colour
02CE	Current auxiliary colour
02CF	Index: parameter save area (while plotting)
02D0	Current character set address page
02D1	Usual character set address page (\$80)
02D2 - 02D3	Pointer: old limit of BASIC memory
02D4	Old screen memory page
02D5	Last scaled X coordinate (0 to 159)
02D6	Last scaled Y coordinate (0 to 159)
02D7	Flag: \$00 = DRAW C,X,Y TO, \$01 = DRAW C TO / current number of out of range coordinates (\$00 = within range)
02D8	Old number of out of range coordinates (\$00 = within range)
02D9	Index: parameter save area (while getting parameters)
02DA - 02FF	(for expansion)
Operating System Vectors	
0300 - 0301	Vector: error message (\$A3FD)
0302 - 0303	Vector: BASIC warm start (\$C483)
0304 - 0305	Vector: lexically analyse BASIC source line (\$A407)
0306 - 0307	Vector: print BASIC tokens in ASCII form (\$A4BA)
0308 - 0309	Vector: start new BASIC statement (\$A504)
030A - 030B	Vector: get and evaluate an expression (\$A52A)
0314 - 0315	Vector: IRQ (\$A372)
0316 - 0317	Vector: BRK instruction (\$A2C2)
0318 - 0319	Vector: NMI (\$FEAD)
031A - 031B	Vector: BASIC OPEN statement (\$F40A)
031C - 031D	Vector: BASIC CLOSE statement (\$F34A)
031E - 031F	Vector: set input (\$F2C7)
0320 - 0321	Vector: set output (\$F309)
0322 - 0323	Vector: restore I/O (\$F3F3)
0324 - 0325	Vector: input a character (\$A395)
0326 - 0327	Vector: output a character (\$A3A6)
0328 - 0329	Vector: test 'STOP' key (\$F770)
032A - 032B	Vector: BASIC GET statement (\$F1F5)
032C - 032D	Vector: abort I/O (\$F3EF)
032E - 032F	Vector: user BRK instruction (\$A2C2)
0330 - 0331	Vector: BASIC LOAD statement (\$F549)
0332 - 0333	Vector: BASIC SAVE statement (\$F685)
0334 - 033B	(for expansion)
Save Area	
033C - 03F8	Save area: parameter passing / pivot coordinates (PAINT)
For CIRCLE	
033C	Index: X or Y
0347 - 0348	Old scaled X coordinate on locus
0349 - 034A	Old scaled Y coordinate on locus
034B - 034C	New scaled X coordinate on locus
034D - 034E	New scaled Y coordinate on locus
034F - 0353	Floating point unit offset X coordinate
0355 - 0359	Floating point unit offset Y coordinate

Instruction Set Summary

Instr	Addressing Mode	Assembler Format	Operation	Op Code		Bytes	Clock Cycles	Status Register - P						Instr
				Hex	Dec			N	V	D	I	Z	C	
ADC	Immediate	ADC #oper	.A + # + C → .A, C	69	105	2	2	N	V	D	I	Z	C	ADC
	Zero Page	ADC addr	.A + [addr] + C → .A, C	65	101	2	3	✓	✓	-	-	✓	✓	
	Zero Page, X	ADC addr, X	.A + [addr + .X] + C → .A, C	75	117	2	4							
	Absolute	ADC ADDR	.A + [ADDR] + C → .A, C	6D	109	3	4							
	Absolute, X	ADC ADDR, X	.A + [ADDR + .X] + C → .A, C	7D	125	3	4*							
	Absolute, Y	ADC ADDR, Y	.A + [ADDR + .Y] + C → .A, C	79	121	3	4*							
	(Indirect, X)	ADC (addr, X)	.A + [[addr + .X + 1, addr + .X]] + C → .A, C	61	97	2	6							
	(Indirect, Y)	ADC (addr, Y)	.A + [[addr + 1, addr] + .Y] + C → .A, C	71	113	2	5*							
AND	Immediate	AND #oper	.A ∧ # → .A	29	41	2	2	N	V	D	I	Z	C	AND
	Zero Page	AND addr	.A ∧ [addr] → .A	25	37	2	3	✓	✓	-	-	✓	✓	
	Zero Page, X	AND addr, X	.A ∧ [addr + .X] → .A	35	53	2	4							
	Absolute	AND ADDR	.A ∧ [ADDR] → .A	2D	45	3	4							
	Absolute, X	AND ADDR, X	.A ∧ [ADDR + .X] → .A	3D	61	3	4*							
	Absolute, Y	AND ADDR, Y	.A ∧ [ADDR + .Y] → .A	39	57	3	4*							
	(Indirect, X)	AND (addr, X)	.A ∧ [[addr + .X + 1, addr + .X]] → .A	21	33	2	6							
	(Indirect, Y)	AND (addr, Y)	.A ∧ [[addr + 1, addr] + .Y] → .A	31	49	2	5*							
ASL	Accumulator	ASL A	.A (←) → .A ; 0 → bit 0, bit 7 → C	0A	10	1	2	N	V	D	I	Z	C	ASL
	Zero Page	ASL addr	[addr] (←) → [addr] "	06	6	2	5	✓	-	-	-	✓	✓	
	Zero Page, X	ASL addr, X	[addr + .X] (←) → [addr + .X] "	16	22	2	6							
	Absolute	ASL ADDR	[ADDR] (←) → [ADDR] "	0E	14	3	6							
	Absolute, X	ASL ADDR, X	[ADDR + .X] (←) → [ADDR + .X] "	1E	30	3	7							
BCC	Relative	BCC oper	Branch on C = 0	90	144	2	2*	N	V	D	I	Z	C	BCC BCS BEQ BNE BMI BPL BVS BVC
BCS	Relative	BCS oper	Branch on C = 1	B0	176	2	2*	-	-	-	-	-	-	
BEQ	Relative	BEQ oper	Branch on Z = 1	F0	240	2	2*	All Branches						
BNE	Relative	BNE oper	Branch on Z = 0	D0	208	2	2*	* - Add 1 if branch to same page						
BMI	Relative	BMI oper	Branch on N = 1	30	48	2	2*	* - Add 2 if branch to diff page						
BPL	Relative	BPL oper	Branch on N = 0	10	16	2	2*							
BVS	Relative	BVS oper	Branch on V = 1	70	112	2	2*							
BVC	Relative	BVC oper	Branch on V = 0	50	80	2	2*							
BIT	Zero Page	BIT addr	.A ∧ [addr] ; bit 7 → N, bit 6 → V	24	36	2	3	N	V	D	I	Z	C	BIT
	Absolute	BIT ADDR	.A ∧ [ADDR]	2C	44	3	4	b ₇	b ₆	-	-	✓	-	
BRK	Implied	BRK 1 → B flag	PC + 2 ↓ P ↓, [FFFE] → PCL, [FFFF] → PCH	00	0	1	7	-	-	-	1	-	-	BRK
CLC	Implied	CLC	0 → C	18	24	1	2	N	V	D	I	Z	C	CLC CLD CLI CLV
	Implied	CLD	0 → D	D8	216	1	2	-	-	-	-	-	0	
	Implied	CLI	0 → I	58	88	1	2	-	-	-	0	-	-	
	Implied	CLV	0 → V	B8	184	1	2	-	0	-	-	-	-	
CMP	Immediate	CMP #oper	.A - #	C9	201	2	2	N	V	D	I	Z	C	CMP
	Zero Page	CMP addr	.A - [addr]	C5	197	2	3	✓	-	-	-	✓	✓	
	Zero Page, X	CMP addr, X	.A - [addr + .X]	D5	213	2	4							
	Absolute	CMP ADDR	.A - [ADDR]	CD	205	3	4							
	Absolute, X	CMP ADDR, X	.A - [ADDR + .X]	DD	221	3	4*							
	Absolute, Y	CMP ADDR, Y	.A - [ADDR + .Y]	D9	217	3	4*							
	(Indirect, X)	CMP (addr, X)	.A - [[addr + .X + 1, addr + .X]]	C1	193	2	6							
	(Indirect, Y)	CMP (addr, Y)	.A - [[addr + 1, addr] + .Y]	D1	209	2	5*							
CPX	Immediate	CPX #oper	.X - #	E0	224	2	2	N	V	D	I	Z	C	CPX
	Zero Page	CPX addr	.X - [addr]	E4	228	2	3	✓	-	-	-	✓	✓	
	Absolute	CPX ADDR	.X - [ADDR]	E4	236	3	4							
CPY	Immediate	CPY #oper	.Y - #	C0	192	2	2	N	V	D	I	Z	C	CPY
	Zero Page	CPY addr	.Y - [addr]	C4	196	2	3	✓	-	-	-	✓	✓	
	Absolute	CPY ADDR	.Y - [ADDR]	CC	204	3	4							
DEC	Zero Page	DEC addr	[addr] - 1 → [addr]	C6	198	2	5	N	V	D	I	Z	C	DEC
	Zero Page, X	DEC addr, X	[addr + .X] - 1 → [addr + .X]	D6	214	2	6	✓	-	-	-	✓	-	
	Absolute	DEC ADDR	[ADDR] - 1 → [ADDR]	CE	206	3	6							
	Absolute, X	DEC ADDR, X	[ADDR + .X] - 1 → [ADDR + .X]	DE	222	3	7							
DEX	Implied	DEX	.X - 1 → .X	CA	202	1	2	N	V	D	I	Z	C	DEX DEY
	Implied	DEY	.Y - 1 → .Y	88	136	1	2	✓	-	-	-	✓	-	
EOR	Immediate	EOR #oper	.A ∪ # → .A	49	73	2	2	N	V	D	I	Z	C	EOR
	Zero Page	EOR addr	.A ∪ [addr] → .A	45	69	2	3	✓	-	-	-	✓	-	
	Zero Page, X	EOR addr, X	.A ∪ [addr + .X] → .A	55	85	2	4							
	Absolute	EOR ADDR	.A ∪ [ADDR] → .A	4D	77	3	4							
	Absolute, X	EOR ADDR, X	.A ∪ [ADDR + .X] → .A	5D	93	3	4*							
	Absolute, Y	EOR ADDR, Y	.A ∪ [ADDR + .Y] → .A	59	89	3	4*							
	(Indirect, X)	EOR (addr, X)	.A ∪ [[addr + .X + 1, addr + .X]] → .A	41	65	2	6							
	(Indirect, Y)	EOR (addr, Y)	.A ∪ [[addr + 1, addr] + .Y] → .A	51	81	2	5*							
INC	Zero Page	INC addr	[addr] + 1 → [addr]	E6	230	2	5	N	V	D	I	Z	C	INC
	Zero Page, X	INC addr, X	[addr + .X] + 1 → [addr + .X]	F6	246	2	6	✓	-	-	-	✓	-	
	Absolute	INC ADDR	[ADDR] + 1 → [ADDR]	EE	238	3	6							
	Absolute, X	INC ADDR, X	[ADDR + .X] + 1 → [ADDR + .X]	FE	254	3	7							
INX	Implied	INX	.X + 1 → .X	E8	232	1	2	N	V	D	I	Z	C	INX INY
	Implied	INY	.Y + 1 → .Y	C8	200	1	2	✓	-	-	-	✓	-	
JMP	Absolute	JMP ADDR	[PC + 1] → PCL, [PC + 2] → PCH	4C	76	3	3	N	V	D	I	Z	C	JMP JSR
	Indirect	JMP (ADDR)	[ADDR] → PCL, [ADDR + 1] → PCH	6C	108	3	5	-	-	-	-	-	-	
	Absolute	JSR ADDR	PC + 2 ↓, [PC + 1] → PCL, [PC + 2] → PCH	20	32	3	6							

Instr	Addressing Mode	Assembler Format	Operation	Op Code		Bytes	Clock Cycles	Status Register						Instr				
				Hex	Dec			N	V	D	I	Z	C					
LDA	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect),Y	LDA #oper LDA addr LDA addr, X LDA ADDR LDA ADDR, X LDA ADDR, Y LDA (addr, X) LDA (addr),Y	# → .A [addr] → .A [addr+.X] → .A [ADDR] → .A [ADDR+.X] → .A [ADDR+.Y] → .A [[addr+.X+1, addr+.X]] → .A [[addr+1, addr+.Y] → .A	A9	169	2	2	N	V	D	I	Z	C	LDA				
				A5	165	2	3	✓	-	-	-	✓	-					
				B5	181	2	4											
				AD	173	3	4											
				BD	189	3	4*											
				B9	185	3	4*											
				A1	161	2	6											
				B1	177	2	5*											
				LDX	Immediate Zero Page Zero Page, Y Absolute Absolute, Y	LDX #oper LDX addr LDX addr, Y LDX ADDR LDX ADDR, Y	# → .X [addr] → .X [addr+.Y] → .X [ADDR] → .X [ADDR+.Y] → .X	A2	162	2	2	N	V		D	I	Z	C
A6	166	2	3					✓	-	-	-	✓	-					
B6	182	2	4															
AE	174	3	4															
BE	190	3	4*															
LDY	Immediate Zero Page Zero Page, X Absolute Absolute, X	LDY #oper LDY addr LDY addr, X LDY ADDR LDY ADDR, X	# → .Y [addr] → .Y [addr+.X] → .Y [ADDR] → .Y [ADDR+.X] → .Y					A0	160	2	2	N	V	D	I	Z	C	LDY
				A4	164	2	3	✓	-	-	-	✓	-					
				B4	180	2	4											
				AC	172	3	4											
				BC	188	3	4*											
				LSR	Accumulator Zero Page Zero Page, X Absolute Absolute, X	LSR A LSR addr LSR addr, Y LSR ADDR LSR ADDR, X	.A (←) → .A ; 0→bit7, bit0→C [addr] (←) → [addr] " [addr+.X] (←) → [addr+.X] " [ADDR] (←) → [ADDR] " [ADDR+.X] (←) → [ADDR+.X] "	4A	74	1	2	N	V	D	I	Z	C	
46	70	2	5					0	-	-	-	✓	✓					
56	86	2	6															
4E	78	3	6															
5E	94	3	7															
NOP	Implied	NOP	No Operation					EA	234	1	2	-	-	-	-	-	NOP	
ORA	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect),Y	ORA #oper ORA addr ORA addr, X ORA ADDR ORA ADDR, X ORA ADDR, Y ORA (addr, X) ORA (addr),Y	.A U # → .A .A U [addr] → .A .A U [addr+.X] → .A .A U [ADDR] → .A .A U [ADDR+.X] → .A .A U [ADDR+.Y] → .A .A U [[addr+.X+1, addr+.X]] → .A .A U [[addr+1, addr+.Y] → .A					09	9	2	2	N	V	D	I	Z	C	ORA
				05	5	2	3	✓	-	-	-	✓	-					
				15	21	2	4											
				0D	13	3	4											
				1D	29	3	4*											
				19	25	3	4*											
				01	1	2	6											
				11	17	2	5*											
				PHA PLA PHP PLP	Implied Implied Implied Implied	PHA PLA PHP PLP	.A ↓, SP - 1 → SP .A ↑, SP + 1 → SP .P ↓, SP - 1 → SP .P ↑, SP + 1 → SP	48 68 08 28	72 104 8 40	1 1 1 1	3 4 3 4	N V D I Z C	- - - - - -	PHA PLA PHP PLP				
				ROL	Accumulator Zero Page Zero Page, X Absolute Absolute, X	ROL A ROL addr ROL addr, X ROL ADDR ROL ADDR, X	.A (←) → .A ; C→bit0, bit7→C [addr] (←) → [addr] " [addr+.X] (←) → [addr+.X] " [ADDR] (←) → [ADDR] " [ADDR+.X] (←) → [ADDR+.X] "	2A	42	1	2	N	V	D	I	Z	C	
26	38	2	5					✓	-	-	-	✓	✓					
36	54	2	6															
2E	46	3	6															
3E	62	3	7															
ROR	Accumulator Zero Page Zero Page, X Absolute Absolute, X	ROR A ROR addr ROR addr, Y ROR ADDR ROR ADDR, X	.A (→) → .A ; C→bit7, bit0→C [addr] (→) → [addr] " [addr+.X] (→) → [addr+.X] " [ADDR] (→) → [ADDR] " [ADDR+.X] (→) → [ADDR+.X] "	6A	106	1	2	N	V	D	I	Z	C	ROR				
				66	102	2	5	✓	-	-	-	✓	✓					
				76	118	2	6											
				6E	110	3	6											
				7E	126	3	7											
				RTI RTS	Implied Implied	RTI RTS	P ↑, PC ↑, SP + 3 → SP, PC + 1 → PC PC ↑, SP + 2 → SP, PC + 1 → PC	40 60	64 96	1 1	6 6	from stack - - - - -	RTI RTS					
SBC	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect),Y	SBC #oper SBC addr SBC addr, X SBC ADDR SBC ADDR, X SBC ADDR, Y SBC (addr, X) SBC (addr),Y	.A - # - C → .A, C ; C = Borrow .A - [addr] - C → .A, C .A - [addr+.X] - C → .A, C .A - [ADDR] - C → .A, C .A - [ADDR+.X] - C → .A, C .A - [ADDR+.Y] - C → .A, C .A - [[addr+.X+1, addr+.X]] - C → .A, C .A - [[addr+1, addr+.Y] - C → .A, C	E9	233	2	2	N	V	D	I	Z	C	SBC				
				E5	229	2	3	✓	✓	-	-	✓	✓					
				F5	245	2	4											
				ED	237	3	4											
				FD	253	3	4*											
				F9	249	3	4*											
				E1	225	2	6											
				F1	241	2	5*											
				SEC SED SEI	Implied Implied Implied	SEC SED SEI	1→C 1→D 1→I	38 F8 78	56 248 120	1 1 1	2 2 2	N V D I Z C	- - - 1 - -		SEC SED SEI			
				STA	Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect),Y	STA addr STA addr, X STA ADDR STA ADDR, X STA ADDR, Y STA (addr, X) STA (addr),Y	.A → [addr] .A → [addr+.X] .A → [ADDR] .A → [ADDR+.X] .A → [ADDR+.Y] .A → [[addr+.X+1, addr+.X]] .A → [[addr+1, addr+.Y]	85	133	2	3	N	V		D	I	Z	C
95	149	2	4					-	-	-	-	-	-					
8D	141	3	4															
9D	157	3	5															
99	153	3	5															
81	129	2	6															
91	145	2	6															
STX	Zero Page Zero Page, Y Absolute	STX addr STX addr, Y STX ADDR	.X → [addr] .X → [addr+.Y] .X → [ADDR]					86	134	2	3	N	V	D	I	Z	C	STX
				96	150	2	4	-	-	-	-	-	-					
				8E	142	3	4											
STY	Zero Page Zero Page, X Absolute	STY addr STY addr, X STY ADDR	.Y → [addr] .Y → [addr+.X] .Y → [ADDR]	84	132	2	3	N	V	D	I	Z	C	STY				
				94	148	2	4	-	-	-	-	-	-					
				8C	140	3	4											
TAX TXA TAY TYA TSX TXS	Implied Implied Implied Implied Implied Implied	TAX TXA TAY TYA TSX TXS	.A → X .X → .A .A → Y .Y → A SP → .X .X → SP	AA 8A A8 98 BA 9A	170 138 168 152 186 154	1 1 1 2 1 1	2 2 2 2 2 2	N V D I Z C	- - - - - -	TAX TXA TAY TYA TSX TXS								

MCS65XX Microprocessor Instruction Set Alphabetic Sequence

ADC	Add memory to accumulator with carry.
AND	AND memory with accumulator.
ASL	Shift left one bit (memory or accumulator).
BCC	Branch on carry clear.
BCS	Branch on carry set.
BEQ	Branch on result zero.
BIT	Test bits in memory with accumulator.
BMI	Branch on result minus.
BNE	Branch on result not zero.
BPL	Branch on result plus.
BRK	Force break.
BVC	Branch on overflow clear.
BVS	Branch on overflow set.
CLC	Clear carry flag.
CLD	Clear decimal mode.
CLI	Clear interrupt disable bit.
CLV	Clear overflow flag.
CMP	Compare memory and accumulator.
CPX	Compare memory and index 'X'.
CPY	Compare memory and index 'Y'.
DEC	Decrement memory by one.
DEX	Decrement index 'X' by one.
DEY	Decrement index 'Y' by one.
EOR	Exclusive-OR memory with accumulator.
INC	Increment memory by one.
INX	Increment index 'X' by one.
INY	Increment index 'Y' by one.
JMP	Jump to new location.
JSR	Jump to new location saving return address.
LDA	Load accumulator with memory.
LDX	Load index 'X' with memory.
LDY	Load index 'Y' with memory.
LSR	Shift right one bit (memory or accumulator).
NOP	No operation.
ORA	OR memory with accumulator.
PHA	Push accumulator on stack.
PHP	Push processor status on stack.
PLA	Pull accumulator from stack.
PLP	Pull processor status from stack.
ROL	Rotate one bit left (memory or accumulator).
ROR	Rotate one bit right (memory or accumulator).
RTI	Return from interrupt.
RTS	Return from subroutine.
SBC	Subtract memory from accumulator with borrow.
SEC	Set carry flag.
SED	Set decimal mode.
SEI	Set interrupt disable status.
STA	Store accumulator in memory.
STX	Store index 'X' in memory.
STY	Store index 'Y' in memory.
TAX	Transfer accumulator to index 'X'.
TAY	Transfer accumulator to index 'Y'.
TSX	Transfer stack pointer to index 'X'.
TXA	Transfer index 'X' to accumulator.
TXS	Transfer index 'X' to stack pointer.
TYA	Transfer index 'Y' to accumulator.

Addressing Modes

Accumulator Addressing - This form of addressing is represented with a one byte instruction, implying an operation on the accumulator.

Immediate Addressing - In immediate addressing, the operand is contained in the second byte of the instruction, with no further memory addressing required.

Absolute Addressing - In absolute addressing, the second byte of the instruction specifies the eight low order bits of the effective address while the third byte specifies the eight high order bits. Thus, the absolute addressing mode allows access to the entire 65k bytes of addressable memory.

Zero Page Addressing - The zero page instructions allow for shorter code and execution times by only fetching the second byte of the instructions and assuming a zero high address byte. Careful use of the zero page can result in significant increase in code efficiency.

Indexed Zero Page Addressing - (X, Y Indexing) - This form of addressing is used in conjunction with the index register and is referred to as "Zero Page, X" or "Zero Page, Y". The effective address is calculated by adding the second byte to the contents of the index register. Since this is a form of "Zero Page" addressing, the content of the second byte references a location in page zero. Additionally due to the "Zero Page" addressing nature of this mode, no carry is added to the high order 8 bits of memory and crossing of page boundaries does not occur.

Indexed Absolute Addressing - (X, Y Indexing) - This form of addressing is used in conjunction with X and Y index register and is referred to as "Absolute, X", and "Absolute, Y". The effective address is formed by adding the contents of X or Y to the address contained in the second and third bytes on the instruction. This mode allows the index register to contain the index or count value and the instruction to contain the base address. This type of indexing allows any location referencing and the index to modify multiple fields resulting in reduced coding and execution time.

Implied Addressing - In the implied addressing mode, the address containing the operand is implicitly stated in the operation code of the instruction.

Relative Addressing - Relative addressing is used only with branch instructions and establishes a destination for the conditional branch. the second byte of the instruction becomes the operand which is an "offset" added to the contents of the lower eight bits of the program counter when the counter is set at the next instruction. The range of the offset is -128 to +127 bytes from the next instruction.

Indexed Indirect Addressing - In indexed indirect addressing (referred to as (Indirect, X)), the second byte of the instruction is added to the contents of the X index register, discarding the carry. The result of the addition points to a memory location on page zero whose contents is the low order eight bits of the effective address. The next memory location in page zero contains the high order eight bits of the effective address. Both memory locations specifying the high and low order bytes of the effective address must be in page zero.

Indirect Indexed Addressing - In indirect indexed addressing (referred to as (Indirect, Y)), the second byte of the instruction points to a memory location in page zero. The contents of this memory location is added to the contents of the Y register, the result being the low order eight bits of the effective address. The carry from this addition is added to the contents of the next page zero memory location, the result being the high order eight bits of the effective address.

Absolute Indirect - The second byte of the instruction contains the low order eight bits of a memory location. The high order eight bits of that memory location is contained in the third byte of the instruction. The contents of the fully specified memory location is the low order byte of the effective address which is loaded into the sixteen bits of the program counter.

BASIC 4.0 / 2.0 Kernal Routines

CBM Label	Address		Operation	Registers In			Registers Out				
	Hex	Dec		.A	.X	.Y	.A	.X	.Y		
CHKIN	FFC6	65478	Open channel for input		LF#		alt.				
CHKOUT	FFC9	65481	Open channel for output		LF#		alt.				
CHRIN	FFCF	65487	Input character from channel				data	alt.			
CHROUT	FFD2	65490	Output character to channel	data							
CLALL	FFE7	65511	Close all channels and files				alt.	alt.			
CLOSE	FFC3	65475	Close a specified logical file	LF#			alt.	alt.	alt.		
CLRCHN	FFCC	65484	Restore default I/O devices				alt.	alt.			
CSYS	FFDE	65502	SYS vector		addr lo	addr hi	alt.	alt.	alt.		
CVERF	FFDB	65499	Verify ram from a device		start lo	start hi		end lo + 1	end hi		
GETIN	FFE4	65508	Get character from current input device				data	alt.	alt.		
LOAD	FFD5	65493	Load ram from a device		start lo	start hi		end lo + 1	end hi		
OPEN	FFC0	65472	Open a logical file				alt.	alt.	alt.		
SAVE	FFD8	65496	Save 'ram' to device	txttab lo	start lo	start hi		end lo + 1	end hi		
STOP	FFE1	65505	Scan stop key depressed	yes: .Z = 1, no .A = last row kybd scan							
UDTIM	FFEA	65514	Increment real time clock					alt.			

alt. = altered

VIC 20 And Commodore 64 Kernal Routines

CBM Label	Address		Operation	Registers In			Registers Out				
	Hex	Dec		.A	.X	.Y	.A	.X	.Y		
ACPTR	FFA5	65445	Input byte from Serial Port				data	alt.			
CHKIN	FFC6	65478	Open channel for input		LF#		alt.				
CHKOUT	FFC9	65481	Open channel for output		LF#		alt.				
CHRIN	FFCF	65487	Input character from channel				data	alt.			
CHROUT	FFD2	65490	Output character to channel	data							
CIOUT	FFA8	65448	Output byte to serial port	data							
CINT	FF81	65409	Initialize screen editor				alt.	alt.	alt.		
CLALL	FFE7	65511	Close all channels and files				alt.	alt.			
CLOSE	FFC3	65475	Close a specified logical file	LF#			alt.	alt.	alt.		
CLRCHN	FFCC	65484	Restore default I/O devices				alt.	alt.			
GETIN	FFE4	65508	Get character from current input device				data	alt.	alt.		
IOBASE	FFF3	65523	Returns base address of I/O devices					addr lo	addr hi		
IOINIT	FF84	65412	Initialize Input/Output				alt.	alt.	alt.		
LISTEN	FFB1	65457	Command devices on the serial bus to listen	DEV#							
LOAD	FFD5	65493	Load (.A = 0) or Verify (.A = 1) 'ram' from a device		start lo	start hi		end lo + 1	end hi		
MEMBOT	FF9C	65436	Read (.C = 1) or Set (.C = 0) the bottom of memory	.C = 0:	bot lo	bot hi	.C = 1:	bot lo	bot hi		
MEMTOP	FF99	65433	Read (.C = 1) or Set (.C = 0) the top of memory	.C = 0:	top lo	top hi	.C = 1:	top lo	top hi		
OPEN	FFC0	65472	Open a logical file				alt.	alt.	alt.		
PLOT	FFF0	65520	Read (.C = 1) or Set (.C = 0) x, y cursor position		row	col		row	col		
RAMTAS	FF87	65415	Init. ram, allocate tape buff, set screen \$0400				alt.	alt.	alt.		
RDTIM	FFDE	65502	Read real time clock				msb	msb2	lsb		
READST	FFB7	65463	Read I/O status word				ST				
RESTOR	FF8A	65418	Restore default I/O vectors				alt.	alt.	alt.		
SAVE	FFD8	65496	Save 'ram' to device	txttab lo	start lo	start hi		end lo + 1	end hi		
SCNKEY	FF9F	65439	Scan keyboard				alt.	alt.	alt.		
SCREEN	FFED	65517	Return screen size in rows & columns					#rows	#cols		
SECOND	FF93	65427	Send secondary address after 'listen'	SA OR \$60							
SETLFS	FFBA	65466	Set logical, first, and second addresses	LF#	DEV#	SA					
SETMSG	FF90	65424	Enable/Disable 'Kernal' messages	.A val: \$40 control msgs on, \$80 error msgs on, \$00 off							
SETNAM	FFBD	65469	Set file name	len	addr lo	addr hi					
SETTIM	FFDB	65499	Set real time clock	msb	msb2	lsb					
SETTMO	FFA2	65442	Set (.A < #128) Reset (.A > #127) IEEE timeout								
STOP	FFE1	65505	Scan stop key depressed	yes: .Z = 1, no .A = last row kybd scan							
TALK	FFB4	65460	Command serial bus device to 'talk'	DEV#							
TKSA	FF96	65430	Send secondary address after 'talk'	SA							
UDTIM	FFEA	65514	Increment real time clock					alt.	alt.		
UNLSN	FFAE	65454	Command serial bus to 'unlisten'				alt.				
UNTLK	FFAB	65451	Command serial bus to 'untalk'				alt.				
VECTOR	FF8D	65421	Store (.C = 1) or Restore (.C = 0) ram vectors	.C = 1:	tabl lo	tabl hi	.C = 0:	tabl lo	tabl hi		

alt. = altered

User Callable ROM Subroutines

Some I/O routines require extra memory set up. See the appropriate Memory Map. Data within parenthesis are for Basic 2.0/4.0 users. (Direct call) indicates no required set up.

#	Entry Point For:								Operation	Registers In			Registers Out		
	2.0		4.0		VIC 20		C64			.A	.X	.Y	.A	.X	.Y
1	C2D8	49880	B350	45904	C3BB	50107	A3BB	41915	Open Up Space In BASIC Text	New:	AryTop Lo		AryTop Hi		Unaltered
2	C328	49960	B3A0	45984	C408	50184	A408	41992	Check Available Memory (called by 1)		(same as above) Start address of move in \$5F, 60 (5C, 5D)				
3	C355	50005	B3CD	46029	C435	50229	A435	42037	?OUT OF MEMORY		(direct call)				
4	C357	50007	BC3F	48191	C437	50231	A437	42039	Send BASIC Error Message	Error #					
5	C389	50057	B3FF	46079	C474	50292	A474	42100	Warm start, BASIC		(direct call)				
6	C399	49960	B40D	46093	C48A	50314	A48A	42122	Main CHRGET entry		(direct call) \$7A = #\$FF, \$7B = #\$01 (\$77, 78);01FF = Basic Inbuf-1				
7	C3AB	50091	B41F	46111	C49C	50220	A49C	42028	Crunch tokens, insert line		Inbuf len.				
8	C439	50233	B4AD	46253	C52A	50474	A52A	42282	Fix chaining, CLR, & READY.		(direct call)				
9	C442	50242	B4B6	46262	C533	50483	A533	42291	Fix chaining		(direct call)				
10	C46F	50287	B4E2	46306	C560	50528	A560	42336	Receive line from keyboard		(direct call) \$7A = #\$FF, \$7B = #\$01				
11	C495	50213	B4FB	46331	C579	50553	A579	42361	Crunch tokens (called by 7)	.X = Inbuf Len. (\$0200,X) = #\$00					
12	C52C	50476	B5A3	46499	C613	50707	A613	42515	Find line in BASIC	StrtBAS Lo	StrtBAS Hi				
13	C55D	50525	B5D4	46548	C642	50754	A642	42562	Do NEW		(direct call)				
14	C572	50546	B5E9	46569	C659	50777	A659	42585	Reset BASIC and do CLR		(direct call)				
15	C575	50549	B5EC	46572	C65E	50782	A65E	42590	Do CLR		(direct call)				
16	C5A7	50599	B622	46626	C68E	50830	A68E	42638	Reset Chrget to Start of BASIC				StrtBAS Hi		
17	C6C4	50884	B74A	46922	C857	51287	A857	43095	Continue BASIC execution [CONT]	CurLin Lo		CurLin Hi			
18	C873	51315	B8F6	47350	C96B	49771	A96B	41579	Get fixed-pt number from BASIC text		Address of text in Chrget ptr; \$7A, 7B (\$77, 78)				
19	C9DE	49886	BADB	47835	CAD3	51923	AA3	43731	Send RETURN, LF if in screen mode				LF (\$0A)		
20	C9E2	49890	BADF	47839	CAD7	51927	AA7	43735	Send RETURN, LINEFEED				LF (\$0A)		
21	CA1C	51740	BB1D	47901	CB1E	51998	AB1E	43806	Print string from A, Y	Addr Lo		Addr Hi			
22	CA22	51746	BB23	47907	CB24	52004	AB24	43812	Print pre-computed string	Length	Addr in \$22,23 (\$1F,20)				
23	CA43	51779	BB44	47940	CB45	52037	AB45	43845	Print '?'		(direct call)				
24	CA45	51781	BB46	47942	CB47	52039	AB47	43847	Print char (output .A to device)	Char			Char		
25	CC9F	52383	BD98	48536	CD9E	52638	AD9E	44446	Evaluate Result string \$0D = #\$FF Expression (\$07) numeric \$0D = #\$00		Address of Expression		Addr Lo		Addr Hi
											In Chrget Pointer		result in Acc#1		
26	CDF8	52728	BEF5	48885	CEFF	52991	AFFD	44797	Check for comma		(direct call)				
27	CDF7	52727	BEF2	48882	CEFA	52986	AFFA	44794	Check for '('		(direct call)				
28	CDF4	52724	BEEF	48879	CEF7	52983	AFF7	44791	Check for ')'		(direct call)				
29	CE03	52739	BF00	48896	CF08	53000	AF08	44808	Send 'SYNTAX ERROR'		(direct call)				
30	CFC9	53193	C187	49543	DOE7	53479	BOE7	45287	Find fl-pt variable, given name				VarAddr Lo		VarAddr Hi
31	D069	53353	C2B9	49849	D185	53637	B185	45445	Bump Variable Addr by 2 (called by 30)	Name in \$45, 46 (\$42, 43)			VarAddr Lo		VarAddr Hi
32	D09A	53290	C2EA	49898	D1BF	53695	B1BF	45503	Float to Fixed conversion in Acc#1		(direct call)				
33	D26D	53869	C4BC	50364	D391	54049	B391	45857	Fixed to Float conversion in Acc#1		(direct call)				
34	D67B	54907	C8D7	51415	D79E	55086	B79E	46894	Get Acc#1 least significant byte to X register				Data		
35	D68F	54927	C8EB	51435	D7B5	55221	B7B5	47029	Evaluate string [VAL]		Address = (Chrget Ptr.)		Fl. Pt. result in Acc#1		
36	D69D	54931	C8EF	51439	D7B9	55225	B7B9	47033	Evaluate string from X, Y (above + 4)		Addr Lo	Addr Hi	Fl. Pt. result in Acc#1		
37	D6C6	54982	C921	49697	D7EB	55275	B7EB	47083	Get two params for POKE, WAIT		Address = (Chrget Ptr.)		X = Pram2, Pram1 in Acc#1 (fxd pt)		
38	D773	55155	C99D	49709	D867	55399	B867	47207	Add (from memory)	Addr Lo		Addr Hi	Fl. Pt. result in Acc#1		
39	D934	53812	CB5E	52062	DA28	55848	BA28	47656	Multiply from memory location	Addr Lo		Addr Hi	Fl. Pt. result in Acc#1		
40	D9EE	53998	CC18	52248	DAE2	56034	BAE2	47842	Multiply Acc#1 by ten				(result in Acc#1)		
41	DAAE	55982	CCD8	52440	DBA2	56226	BBA2	48034	Unpack memory variable to Acc#1	Addr Lo		Addr Hi			
42	DAE3	56035	CD0D	52493	DBD7	56279	BBD7	48087	Copy Acc#1 to (X,Y) Location	Addr Lo	Addr Hi				
43	DB08	56072	CD32	52530	DBFC	56316	BBFC	48124	Move Acc#2 to Acc#1		(direct call)				
44	DB18	56088	CD42	52546	DC0C	56332	BC0C	48140	Move Rounded Acc#1 to Acc#2		(direct call)				
45	DB1D	56093	CD45	52549	DC0F	56335	BC0F	48143	Move Un-Rounded Acc#1 to Acc#2		(direct call)				
46	DB27	56103	CD51	52561	DC1B	56347	BC1B	48155	Round Acc.#1		(direct call)				
47	DCD9	56537	CF83	53123	DDCD	56781	BDCD	48589	Print fixed-point value	Value Hi	Value Lo				
48	DCE3	56547	CF8D	53133	DDD7	56791	BDD7	48599	Print floating-point value in Acc#1		(direct call)				
49	DCE9	56553	CF93	53027	DDDD	56797	BDDD	48605	Convert num to strng at \$0100 (call'd by 47)	#\$00		#\$01			
50	FD11	64785	D472	54386	n/a	n/a	n/a	n/a	Entry to M.L.M.		(direct call)				
51	E3D8	58328	E202	57858	E742	59202	E716	59158	Print a character	Char					
52	F156	61782	F185	61829	F1E6	61926	F12F	61743	Print system message			Offset			
53	F0B6	61622	F0D2	61650	EE14	60948	ED09	60681	Send 'talk' to IEEE/Serial	Dev #					
54	F0BA	61626	F0D5	61653	EE17	60951	ED0C	60684	Send 'listen' to IEEE/Serial	Dev #					
55	F128	61736	F143	61763	FF93	65427	FF93	65427	Send secondary address	SA OR \$60					
56	F16F	61807	F19E	61742	EEE4	61156	ED40	60736	Send char to IEEE/Serial	Char					
57	F17F	61823	F1AE	61870	EEF6	61174	EDEF	60911	Send 'untalk'		(direct call)				
58	F183	61827	F1B9	61881	EF04	61188	EDFE	60926	Send 'unlisten'		(direct call)				

#	Entry Point For:								Operation	Registers Out			
	2.0		4.0		VIC 20		C64			.A	.Y	.Z	.C
59	F18C	61836	F1C0	61888	EF19	61209	EE13	60947	Input from IEEE/Serial			Data	
60	F2A9	62121	F2DD	62173	F34A	62282	F291	61985	Close logical file (kernal rtn)	LF #			
61	F301	62209	F335	62261	F770	63344	F6ED	63213	Check for STOP key			Z flag = 1 if pressed	
62	F322	62242	F356	62294	F542	62786	F49E	62510	LOAD subroutine	#\$00	Start Lo	Start Hi	
63	F40A	62474	F449	62537	F647	63047	F5AF	62895	Print SEARCHING...		(direct call)		
64	F41D	62493	F45C	62556	F659	63065	F5C1	62913	Print file name		(direct call)		
65	F494	62500	F4D3	62675	F867	63591	F7EA	63466	Find specific tape header block	Len	Pointer to string in \$BB, BC (same for 2/4.0)		
66	F5A6	62886	F5E5	62949	F7AF	63407	F72D	63277	Find any tape header block		(direct call)		
67	F812	63506	F857	63575	F894	63524	F817	63511	Press PLAY...; wait		(direct call)		
68	F855	63573	F89A	63530	F8C0	63680	F841	63553	Read tape to buffer		(direct call)		
69	F85E	63582	F8A3	63651	F8C6	63686	F847	63559	Read tape		(direct call)		
70	F886	63622	F8CB	63691	F8E3	63715	F864	63588	Write tape from buffer				
71	F88E	63630	F8D3	63699	F8E8	63720	F869	63593	Write tape, leader length in A	Ldr Len.			
72	FB76	64374	FB76	64443	FCF6	64758	FB8E	64398	Reset tape I/O		(direct call)		
73	FC9B	64555	FCE0	64736	FCF9	64761	FCBD	64701	Set interrupt vector		(direct call)		

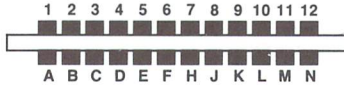
BASIC Keyword Tokens and Entry Points

Keyword	Token		ROM Entry Point							
	Hex	Dec	BASIC 2.0		BASIC 4.0		VIC 20	C64		
ABS	B6	182	DB64	56164	CD8E	52622	DC58	59408	BC58	48216
AND	AF	175	CECB	52939	C089	49289	CFE9	53225	AFE9	45033
APPEND**	D4	212			FFAB	65451				
ASC	C6	198	D665	54885	C8C1	51393	D78B	55179	B78B	46987
ATN	C1	193	E08C	57484	D32C	54060	E30B	58123	E30E	58126
BACKUP**	D2	210			FFA5	65445				
CATALOG**	DA	215			FFB4	65460				
CHR	C7	199	D5C6	54726	C822	51234	D6E6	55020	B6EC	46828
CLOSE*	A0	160	FFC3	65475	FFC3	65475	FFC3	65475	FFC3	65475
CLR	9C	156	C577	50551	B5EE	46574	C65E	50782	A65E	42590
CMD	9D	157	C991	51601	BA8E	47758	CA86	51846	AA86	43654
COLLECT**	D1	209			FFA2	65442				
CONCAT**	CC	204			FF93	65427				
CONT	9A	154	C76B	51051	B7EE	47086	C857	51287	A857	43095
COPY**	D3	211			FFA8	65448				
COS	BE	190	DFD8	57304	D282	53890	E261	57953	E264	57956
DATA	83	131	C800	51200	B883	47235	C858	51448	A8F8	43256
DCLOSE**	CE	206			FF99	65433				
DEF	96	150	D28D	53901	C4DC	50396	D3B3	54195	B3B3	46003
DIM	86	134	CF63	53091	C121	49441	D081	53377	B081	45185
DIRECTORY**	DA	218			FFB4	65460				
DLOAD**	CD	205			FF96	65430				
DSAVE**	D5	213			FFAE	65454				
END	80	128	C741	51009	B7C8	47048	C831	51249	A831	43057
EXP	BD	189	DEDA	57050	D184	53636	DFED	57325	BFED	49133
FN	A5	165	D2CE	53966	C51D	50461	D3F4	54260	B3F4	46068
FOR	81	129	C658	50776	B6DE	46814	C742	51010	A742	42818
FRE	B8	184	D259	53849	C4A8	50344	D37D	54141	B37D	45949
GET*	A1	161	FFE4	65508	FFE4	65508	FFE4	65508	FFE4	65508
GOSUB	8D	141	C790	51088	B813	47123	C883	51331	A883	43139
GOTO	89	137	C7AD	51117	B830	47152	C8A0	51360	A8A0	43168
HEADER**	D0	208			FF9F	65439				
IF	8B	139	C830	51248	B8B3	47283	C928	51496	A928	43304
INPUT*	85	133	FFCF	65487	FFCF	65487	FFCF	65487	FFCF	65487
INPUT#	84	132	CAA7	51879	BBA4	48036	CBA5	52133	ABA5	43941
INT	B5	181	DBD8	56280	CE02	52738	DCCC	56524	BCCC	48332
LEFT	C8	200	D5DA	54746	C836	51254	D700	55040	B700	46848
LEN	C3	195	D656	54870	C8B2	51378	D77C	55164	B77C	46972
LET	88	136	C8AD	51373	B930	47408	C9A5	51621	A9A5	43429

Token	BASIC 2.0		BASIC 4.0		VIC 20		C64				
	LIST	9B	155	C5B5	50613	B630	46640	C69C	50844	A69C	42652
LOAD*	93	147	FFD5	65493	FFD5	65493	FFD5	65493	FFD5	65493	
LOG	BC	188	D8F6	55542	CB20	52000	D9EA	55786	B9EA	47594	
MID	CA	202	D611	54801	C86D	51309	D737	55095	B737	46903	
NEW	A2	162	C55B	50523	B5D2	46546	C642	50754	A642	42562	
NEXT	82	130	CC20	52256	BD19	48409	CD1E	52510	AD1E	44318	
NOT	A8	168	CDCD	52687	BECC	48844	CED4	52948	AED4	44756	
ON	91	145	C853	51283	B8D6	47318	C94B	51531	A94B	43339	
OPEN*	9F	159	FFC0	65472	FFC0	65472	FFC0	65472	FFC0	65472	
OR	B0	176	CEC8	52936	C086	49286	CFE6	53222	AFE6	45030	
PEEK	C2	194	D6E8	55016	C943	51523	D80D	55309	B80D	47117	
POKE	97	151	D707	55047	C95A	51546	D824	55332	B824	47140	
POS	B9	185	D27A	53882	C4C9	50377	D39E	54174	B39E	45982	
PRINT*	99	153	FFD2	65490	FFD2	65490	FFD2	65490	FFD2	65490	
PRINT#	98	152	C98B	51595	BA88	47752	CA80	51840	AA80	43648	
READ	87	135	CB07	51975	BC02	48130	CC06	52230	AC06	44038	
RECORD**	CF	207			FF9C	65436					
REM	8F	143	C843	51267	B8C6	47302	C93B	51515	A93B	43323	
RENAME**	D8	216			FFB7	65463					
RESTORE	8C	140	C730	50992	B7B7	47031	C81D	51229	A81D	43037	
RETURN	83	132	C7DA	51162	B85D	47197	C8D2	51410	A8D2	43218	
RIGHT	C9	201	D606	54790	C862	51298	D72C	55084	B72C	46892	
RND	BB	187	DF7F	57215	D229	53801	E094	57492	E097	57495	
RUN	8A	138	C785	51077	B808	47112	C871	51313	A871	43121	
SAVE*	94	148	FFD8	65496	FFD8	65496	FFD8	65496	FFD8	65496	
SCRATCH**	D9	217			FFBA	65466					
SGN	B4	180	BD45	56133	CD6F	52591	DC39	56377	BC39	48185	
SIN	BF	191	DFDF	57311	D289	53897	E268	57960	E26B	57963	
SPC(A6	166	C9FC	51708	BAFD	47869	CAF8	51960	AAF8	43768	
SQR	BA	186	DE5E	56926	D108	53512	DF71	57201	BF71	49009	
STEP	A9	169	C6AB	50859	B731	46897	C795	51093	A795	42901	
STOP	90	144	C73F	51007	B7C6	47046	C82F	51247	A82F	43055	
STR	C4	196	D33F	54079	C58E	50574	D465	54373	B465	46181	
SYS*	9E	158	F684	63108	F6C3	63171	E127	57639	E12A	57642	
TAB(A3	163	C9FC	51708	BAFD	47869	CAF8	51960	AAF8	43768	
TAN	C0	192	E028	57384	D2D2	53970	E2B1	58033	E2B4	58036	
USR	B7	183	All ROMs Jump to Location 0000 (thru USR Jump Vector)								
VAL	C5	197	D687	54919	C8E3	51427	D7AD	55213	B7AD	47021	
VERIFY*	95	149	FFDB	65499	FFDB	65499	FFDB	65499	FFDB	65499	
WAIT	92	146	D710	55056	C963	51555	D82D	55341	B82D	47149	

* Kernal Routine / ** BASIC 4.0 Kernal Routine

RS 232 User Port Lines



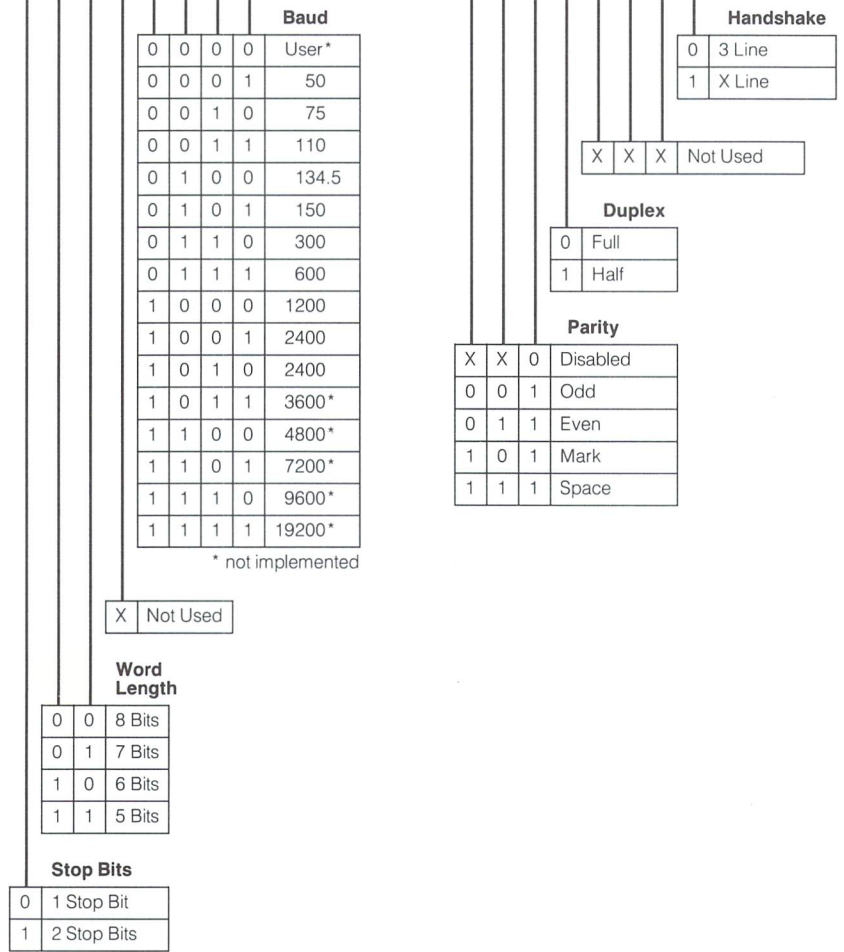
VIC 20 RS 232 is controlled by VIA 1 (6522) at \$9110
 C64 RS 232 is controlled by CIA 2 (6526) at \$DD00

Pin#	Chip	Description	Abv	Dir.	Modes
A	GND	Protective Ground	GND		1 2
B	FLAG2	Received Data	S _{in}	IN	1 2
C	PB0	Received Data	S _{in}	IN	1 2
D	PB1	Request to Send	RTS	OUT	1* 2
E	PB2	Data Terminal Ready	DTR	OUT	1* 2
F	PB3	Ring Indicator	RI	IN	3
H	PB4	Received line Signal	DCD	IN	2
J	PB5	Unassigned		IN	3
K	PB6	Clear To Send	CTS	IN	2
L	PB7	Data Set Ready	DSR	IN	2
M	PA2	Transmitted Data	S _{out}	OUT	1 2
N	GND	Signal Ground	GND		1 2 3

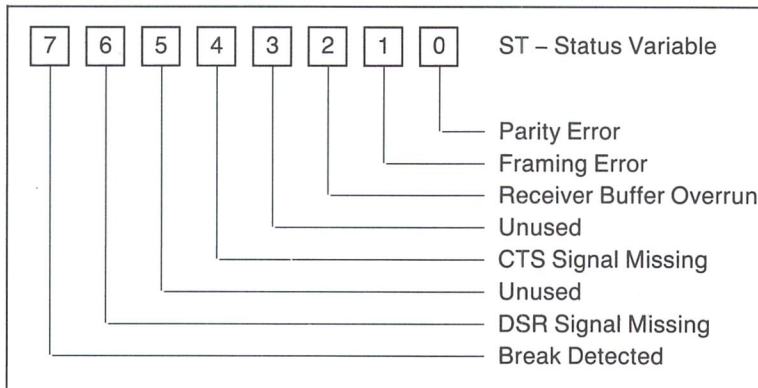
Available Modes

- 1) 3-Line interface (S_{in}, S_{out}, GND)
 - 2) X-Line interface
 - 3) User available only (unused in code)
- * these lines are held high during 3-line mode.

Control Register
 OPEN LF, 2, SA, CHR\$((7 6 5 4 3 2 1 0)) + CHR\$((7 6 5 4 3 2 1 0))



VIC 20 / C64 RS 232 Status



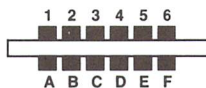
Notes:

- If the LF# is 128 or greater, a Line Feed will be sent after each Carriage Return
- The Secondary Address SA does not affect RS 232 operation
- Before Closing the channel, check output buffer for data with:
 VIC 20 : 100 IF ST = 0 AND (PEEK(37151) AND 64) = 64 THEN 100
 110 CLOSE LF
 C64 : 100 SX = ST : IF (SX = 0 OR SX = 8) THEN 100
 110 CLOSE LF

IEEE 488 Bus Signals

Manager	ATN	Attention	The controller (PET/CBM/B) sets this signal low while it is sending commands on the data bus. When ATN is low, only peripheral addresses and control messages are on the data bus. When ATN is high, only previously assigned devices can transfer data.
Transfer	DAV	Data Valid	When DAV is low, this signifies that data is valid on data bus.
Manager	EOI	End or Identify	When the last byte of data is being transferred, the talker has the option of setting EOI low. The controller always sets EOI low while the last data byte is being transferred from the controller.
Manager	IFC	Interface Clear	The controller sends its internal reset signal as IFC low (true) to initialize all devices to the idle state. When the controller is switched on or reset, IFC goes low for about 100 milliseconds.
Transfer	NDAC	Data Not Accepted	This signal is held low (true) by the listener while reading. When the data byte has been read, the listener sets NDAC high. This signals the talker that data has been accepted.
Transfer	NRFD	Not Ready for Data	When NRFD is low (true), one or more listeners are not ready for the next byte of data. When all devices are ready, NRFD goes high.
Manager	SRQ	Service Request	Not implemented in BASIC, but available to the user.
Manager	REN	Remote Enable	REN is held low by the bus controller. The PET/CBM has a pin grounded that keeps REN permanently low.
Data	D101-8	Data Input/Output Lines 1-8	These signals represent the bits of information on the data bus. When a D10 signal is low, it represents 1 and when high 0.
General	GND	Ground	Ground connections: There are six control and management signal ground returns, one data signal ground return and one chassis shield ground lead.

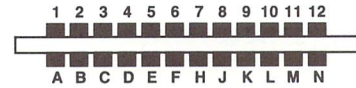
Cassette Port



Pin#	Name	Description
A-1	GND	Digital Ground
B-2	+5V	+5 Volts to operate cassette circuitry only
C-3	Motor	Computer controlled +6V for cassette motor
D-4	Read	Read line from cassette
E-5	Write	Write line cassette
F-6	Sense	Monitors closure of any locking type cassette switch

Note: Upper and Lower cassette pins are shorted

IEEE Port Pinouts



Pin #	Pin# *	Mnemonic	Definition
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 Identify
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 (IEEE 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 (DIO1-8)

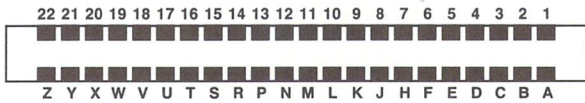
* Pin Numbers for Standard IEEE Cable Connector

PET/CBM User Port



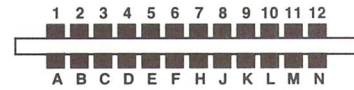
Pin#	Function	Description
1	Ground	System Ground
2	TV Video	Video Out for external displays
3	SRQ	Connected to IEEE SRQ
4	EOI	Connected to IEEE EOI
5	Diag Sense	Held low causes power up to Diagnostic routines
6	READ 1	Connected to cassette 1 read line
7	READ 2	Connected to cassette 2 read line
8	Write	Diagnostic tape write verify
9	Vert	TV Vertical for external displays
10	Horiz	TV Horizontal for external displays
11	GND	
12	GND	
A	GND	
B	CA1	Edge sensitive input of 6522 VIA
C	PB0	PB0-7 are independently programmable for Input or Output
D	PB1	
E	PB2	
F	PB3	
H	PB4	
J	PB5	
K	PB6	
L	PB7	
M	CB2	Special I/O pin of VIA
N	GND	Digital Ground

VIC 20 Expansion Port



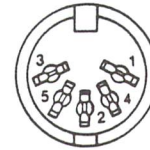
Pin#	Name	Description
1	GND	System ground
2	CD0	Data bus bit 0
3	CD1	Data bus bit 1
4	CD2	Data bus bit 2
5	CD3	Data bus bit 3
6	CD4	Data bus bit 4
7	CD5	Data bus bit 5
8	CD6	Data bus bit 6
9	CD7	Data bus bit 7
10	BLK1	8k decoded RAM/ROM block 1 @ \$2000 (active low)
11	BLK2	8k decoded RAM/ROM block 2 @ \$4000 (active low)
12	BLK3	8k decoded RAM/ROM block 3 @ \$6000 (active low)
13	BLK5	8k decoded ROM block 5 @ \$A000 (active low)
14	RAM1	1k decoded RAM block @ \$0400 (active low)
15	RAM2	1k decoded RAM block @ \$0800 (active low)
16	RAM3	1k decoded RAM block @ \$0C00 (active low)
17	V R/W	Read/Write line from VIC Chip (high-read, low-write)
18	C R/W	Read/Write line from CPU (high-read, low-write)
19	IRQ	Interrupt Request line to 6502 (active low)
20	NC	
21	+5v	
22	GND	
A	GND	
b	CA0	Address bus bit 0
C	CA1	Address bus bit 1
D	CA2	Address bus bit 2
E	CA3	Address bus bit 3
F	CA4	Address bus bit 4
H	CA5	Address bus bit 5
J	CA6	Address bus bit 6
K	CA7	Address bus bit 7
L	CA8	Address bus bit 8
M	CA9	Address bus bit 9
N	CA10	Address bus bit 10
P	CA11	Address bus bit 11
R	CA12	Address bus bit 12
S	CA13	Address bus bit 13
T	I/O 2	I/O block 2 (located at \$9600)
U	I/O 3	I/O block 3 (located at \$9C00)
V	Φ02	Phase 2 system clock
W	NMI	6502 Non-Maskable Interrupt (active low)
X	RESET	6502. Reset pin (active low)
Y	NC	
Z	GND	

VIC 20 User Port



Pin#	Function	Description
1	Ground	System Ground
2	+5V	(100 ma maximum)
3	RESET	Cold Start. Memory is destroyed
4	JOY 0	Joystick Switch 0
5	JOY 1	Joystick Switch 1
6	JOY 2	Joystick Switch 2
7	PEN	Light Pen Input. Also Joystick Fire Button
8	SENSE	Cassette Switch sense line
9	Serial ATN	Connected to Serial Bus ATN Line
10	9 VAC + Phase	Transformer output (50 ma. maximum)
11	GND	
12	GND	
A	GND	
B	CB1	
C	PB0	PB0-7 are independently programmable for Input or Output
D	PB1	
E	PB2	
F	PB3	
H	PB4	
J	PB5	
K	PB6	
L	PB7	
M	CB2	
N	GND	Special I/O pin of VIA

VIC 20 Audio/Video Port



Pin#	Name	Description	Colour
1	+5V	10 ma. maximum	Red
2	GND	System Ground	-
3	AUD	Audio Out	Grey
4	VID L	Video Low	Black
5	VID H	Video High	White

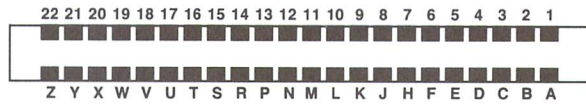
Colour refers to Radio Shack Part# 42-2394

VIC 20 / Commodore 64 Serial Port

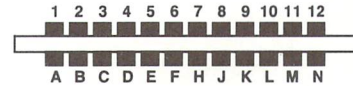


Pin#	Name	Description
1	SRQ	Serial SRQ in (active low)
2	GND	System Ground
3	ATN	Serial ATN In/Out
4	CLK	Serial Clock In/Out
5	DATA	Serial Data In/Out
6	RESET	Resets all devices on Serial bus (active low)

Commodore 64 Expansion Port



Commodore 64 User Port



Name	Pin#	Description
1	GND	System Ground.
2	+5 VDC	Total User Port and Cartridge devices
3	+5 VDC	can draw no more than 450ma.
4	IRQ	Interrupt Request line to 6502 (active low).
5	R/W	Read/Write.
6	Dot	
	Clock	8.18 MHz video dot clock.
7	I/O 1	I/O Block 1 @ \$DE00-\$DEFF (active low) unbuffered I/O.
8	GAME	Active low TTL input.
9	EXROM	Active low TTL input.
10	I/O 2	I/O Block 2 @ \$DF00-\$DFFF (active low) buffered TTL output.
11	ROM L	8K decoded RAM/ROM block @ \$8000 (active low) buffered TTL output.
12	BA	Bus Available signal from the VIC II chip - unbuffered - 1 is maximum load.
13	DMA	Direct Memory Access request line (active low input) is TTL input.
14	D7	Data bus bit 7 *
15	D6	Data bus bit 6 *
16	D5	Data bus bit 5 *
17	D4	Data bus bit 4 *
18	D3	Data bus bit 3 *
19	D2	Data bus bit 2 *
20	D1	Data bus bit 1 *
21	D0	Data bus bit 0 *
21	GND	System ground.
A	GND	System Ground
B	ROM H	8K decoded RAM/ROM Block @ \$E000 buffered.
C	RESET	6502 RESET pin (active low) buffered TTL out/unbuffered in.
D	NMI	6502 Non-Maskable Interrupt (active low) buffered TTL out, unbuffered in.
E	Φ2	Phase 2 system clock.
F	A15	Address bus bit 15 *
H	A14	Address bus bit 14 *
J	A13	Address bus bit 13 *
K	A12	Address bus bit 12 *
L	A11	Address bus bit 11 *
M	A10	Address bus bit 10 *
N	A9	Address bus bit 9 *
P	A8	Address bus bit 8 *
R	A7	Address bus bit 7 *
S	A6	Address bus bit 6 *
T	A5	Address bus bit 5 *
U	A4	Address bus bit 4 *
V	A3	Address bus bit 3 *
W	A2	Address bus bit 2 *
X	A1	Address bus bit 1 *
Y	A0	Address bus bit 0 *
Z	GND	System Ground

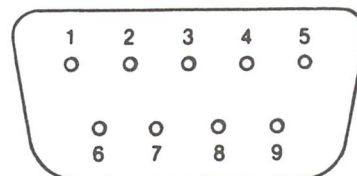
* = Unbuffered, 1 is TTL load max.

Pin#	Function	Description
1	Ground	System Ground
2	+5V	(100 ma maximum)
3	RESET	Cold Start. Memory is NOT destroyed
4	CNT1	Serial Port counter from CIA #1
5	SP1	Serial Port from CIA #1
6	CNT2	Serial Port counter from CIA #2
7	SP2	Serial Port from CIA #2
8	PC2	Handshaking line from CIA #2
9	Serial ATN	Connected to Serial Bus ATN Line
10	9 VAC +Phase	Transformer output (50 ma. maximum)
11	9 VAC -Phase	Transformer output (50 ma. maximum)
12	GND	
A	GND	
B	FLAG2	
C	PB0	PB0-7 are independently programmable for input or output
D	PB1	
E	PB2	
F	PB3	
H	PB4	
J	PB5	
K	PB6	
L	PB7	
M	PA2	Special I/O pin of CIA
N	GND	

Commodore 64 Audio/Video Port

Pin#	Name	Description
1	LUM	Luminance
2	GND	System Ground
3	AUD	Audio Out
4	COMP	Composite Video
5	JACK	Audio In
6	CHR	Chroma out
7	N/C	No connection
8	N/C	No connection

VIC 20 / Commodore 64 Joystick Ports



Pin#	Name	Description
1	JOY 0	
2	JOY 1	
3	JOY 2	
4	JOY 3	
5	POT Y	
6	FIRE	Also the Light Pen input. (C64 port 1 only)
7	+5V	100 ma. maximum
8	GND	System Ground
9	POT X	

Note: See Memory Map for reading Joystick Ports

Model	D9090	D9060	8250	8050	4040	2031	1541
Drives per Head	1	1	2	2	2	1	1
Heads per Drive	6	4	2	1	1	1	1
Formatted Storage							
Capacity per Unit	7.47 MB	4.98 MB	2.12 MB	1.05 MB	340 KB	170KB	170KB
Max Sequential Files/Drive	7.41 MB	4.94 MB	1.05 MB	521 KB	168 KB	168 KB	168KB
Max Relative Files/Drive	7.35 MB	4.90 MB	1.04 MB	183 KB	167 KB	167 KB	167KB
Disk System Buffer	4 KB	4 KB	4 KB	4 KB	4 KB	2 KB	2KB
Disk Formats							
Cylinders (Tracks)	153	153	77	77	35	35	35
Sectors per Cylinder	128	192	-	-	-	-	-
Sectors per Track	32	32	23-29	23-29	17-21	17-21	17-21
Bytes per Sector	256	256	256	256	256	256	256
Blocks Free	29162	19442	8266	4104	1328	664	664
Transfer Rates (bytes per second)							
Internal to Unit	5 MB	5 MB	40 KB	40 KB	40 KB	40 KB	-
IEEE-488 Bus	1.2 KB	1.2 KB	1.2 KB	1.2 KB	1.2 KB	1.2 KB	-
Access Times (milli-seconds)							
Track-To-Track	3	3	5	*	30	30	30
Average Track	153	153	125	**	360	360	360
Head Settling Time	15	15	-	-	-	-	-
Average Latency	8.34	8.34	100	100	100	100	100
RPM	3600	3600	300	300	300	300	300
* Track-To-Track: Micropolis 8050 = 30 ms. Tandon 8050 = 5 ms. ** Average Track: Micropolis 8050 = 750 ms. Tandon 8050 = 125 ms.							
Physical Dimensions							
Height (inches)	5.75	5.75	7.0	7.0	7.0	5.5	3.0
Width (inches)	8.25	8.25	15.0	15.0	15.0	8.0	7.0
Depth (inches)	15.25	15.25	13.75	13.75	13.75	14.25	13.0
Weight (pounds)	21	21	28	28	28	20	10
Electrical							
Power (Watts)	200	200	60	50	50	40	35
Voltage (all models)	110 - 120 VAC. 60 Hz						

Disk Utility-Command Set

Command	Abbreviation	Format
Block-Read	B-R	"B-R: " ch;dr;t;s
Block-Write	B-W	"B-W: " ch;dr;t;s
Block-Execute	B-E	"B-E: " ch;dr;t;s
Buffer-Pointer	B-P	"B-P: " ch;p
Block-Allocate	B-A	"B-A: " dr;t;s
Block-Free	B-F	"B-F: " dr;t;s
Memory-Write	M-W	"M-W " adL/adH/nc/data
Memory-Read	M-R	"M-R " adl/adh
Memory-Execute	M-E	"M-E " adl/adh
User Command	U	"ux:ch;dr;t;s

CH	The channel number in DOS: identical to the Secondary Address in the associated OPEN statement
DR	The Drive number: 0 (or 1 floppy dual drives)
T	The Track number: 1 through 154 (depending on the model#)
S	Sector number : 0 through 112 (depending on the model#)
P	The pointer Position for the buffer pointer
ADL	The Low byte of the Address (use CHR\$(ADL))
ADH	The High byte of the Address (use CHR\$(ADL))
NC	The Number of Characters: 1 through 34
DATA	The actual data in hexadecimal. This is transmitted by using the CHR\$ function, ie. CHR\$(17) would send the decimal equivalent of hexadecimal 11
X	The index to the user table
PARMS	The Parameters associated with the U command (optional)

Sector Distribution By Track

Track Number	Number of Sectors		
	4040	2031	1541
1 - 17	21	21	21
18 - 24	19	19	19
25 - 30	18	18	18
31 - 35	17	17	17

Track Number	8050	8250
1 - 39	29	29
40 - 53	27	27
54 - 64	25	25
65 - 77	23	23
78 - 116		29
117 - 130		27
131 - 141		25
142 - 154		23

D9060/D9090 - 153 tracks per recording surface (4 on D9060 and 6 on the D9090) with 32 sectors per track

User Command Jump Table

Standard Syntax	Alternate Syntax	Function
U1	UA	Block-Read replacement
U2	UB	Block-Write replacement
U3	UC	Jump to \$1300
U4	UD	Jump to \$1303
U5	UE	Jump to \$1306
U6	UF	Jump to \$1309
U7	UG	Jump to \$130C
U8	UH	Jump to \$130F
U9	UI	Jump to \$10F0 (NMI)
U:	UJ	Power-Up vector (reset)

4040, 2031, and 1541 BAM Format - Track 18 Sector 00					
Byte#	Description	Data			
0-1	Track-Sector of first Directory block	18-00			
2	ASCII 'a' Identifies DOS 2.6 format	65			
3	Reserved for future DOS use	00			
4-143	Bit map of available blocks	tracks 1-35			
8050 BAM Format					
Byte#	Description	Data			
		BAM 1 Tr38 / Sc00	BAM 2 Tr38 / Sc03		
0-1	Track-Sector of next BAM block	38-03	39-01		
2	ASCII 'c' Identifies DOS 2.5 format	67	67		
3	Reserved for future DOS use	00	00		
4	Lowest track # mapped in this BAM block	01	51		
5	Highest track # (+ 1) mapped in this BAM block	51	78		
6	Number of unused blocks on track:	1	51		
7-10	Bit map of available blocks on track:	1	51		
11-255	(BAM 2: 11-140)Bit map of available blocks on tracks:	2-50	52-77		
8250 BAM Format					
Byte#	Description	Data			
		BAM 1 Tr38 / Sc00	BAM 2 Tr38 / Sc03	BAM 3 Tr38 / Sc06	BAM 4 Tr38 / Sc09
0-1	Track-Sector of next BAM block	38-03	38-06	38-09	38-01 (Dir)
2	ASCII 'c' Identifies DOS 2.7 format	67	67	67	67
3	Reserved for future DOS use	00	00	00	00
4	Lowest track # mapped in first BAM block	01	51	101	151
5	Highest track # (+ 1) mapped in first BAM block	51	101	151	155
6	number of unused blocks on track:	1	51	101	151
7-10	bit map of available blocks on track:	1	51	101	151
11-255	(BAM 4: 11-25)Bit map of available blocks on tracks:	2-50	52-100	102-150	152-154
D9060 / D9090 BAM Format - Track 1 Sector 0 (normal location)					
Byte#	Description	Data			
0-1	Track-Sector pointer to next BAM block	\$FFFF = last			
2-3	Track-Sector pointer to previous BAM block	\$FFFF = first			
4	Lowest track # mapped in this BAM block				
5	Highest track # (+ 1) mapped in this BAM block				
6	Number of blocks unused on this track				
7-10	Bit map of available blocks on this track				
11-255	Bit map of the next 49 tracks				

Directory Format

2031, 4040, 1541 Directory Header - Track 18 Sector 00		
Byte#	Data	Description
1-143		Reserved for 2031 BAM
144-161		Diskette name, padded with shifted spaces
162-163		Diskette ID number
164	160	Shifted space
165-166	50, 65	ASCII '2a' identifies DOS version and format
167-170	160	Shifted spaces
171-255	00	Not used
8050, 8250 Directory Header - Track 39 Sector 00		
Byte#	Data	Description
0-1	38, 00	Track-Sector to first BAM block
2	67	ASCII 'c' identifies DOS 2.5 format
3	00	reserved for future DOS use
4-5		Not used
6-21		Diskette name, padded with shifted spaces
22-23	160	Shifted spaces
24-25		Diskette ID number
26	160	Shifted space
27-28	50, 67	ASCII '2c' identifies DOS version and format
29-32	160	Shifted spaces
33-255	00	Not used
D9060 / D9090 Directory Header - Track 0 Sector 0		
Byte#	Data	Description
0-1		Track-Sector pointer to bad track and sector list
2-3	00,255	Identifies DOS 3.0 format
4-5	76, 00	Track-Sector of first directory block
6-7	00, 00	Not used
8-9	01, 00	Track-Sector of first BAM block

2031 Directory Blocks - Track 18 Sector 01 through 18
 4040 Directory Blocks - Track 18 Sector 01 through 18
 8050 Directory Blocks - Track 39 Sector 01 through 29
 8250 Directory Blocks - Track 39 Sector 01 through 29
 D9060 / D9090 Directory Blocks - Starting on cylinder 76, uses all Tracks - Sectors 00 through 31, then expands to additional blocks as required, providing 'unlimited' Directory size.

Byte#	Description
0-1	Track-sector pointer to next directory block
2	File type
3-4	Track-sector pointer to first file block
5-20	File name, padded with shifted spaces
21-22	Track-sector of first side sector if RELative file
23	Record length if relative file
24-27	Reserved for future file information
28-29	Track-sector pointer for replacement
30-31	Number of blocks used by the file
32-255	Seven more 32-byte file entries (same as 2-31 above, plus two additional unused bytes)

Additional Notes	
1	32 bytes per file entry, except the first entry is 30 bytes
2	Total of eight (8) file entries per directory block
3	File types are: Scratched Files \$00 Sequential Files \$01 Program Files \$02 User-Defined \$03 Relative Record \$04
4	File type codes are OR'ed with \$80 when file is properly closed
5	Track value of 00 in byte zero indicates the last used block in the directory. Sector value then shows next byte to use

Commodore Related Programming Books

Title	Author	Publisher
SuperPET Related Books. SuperPET System Overview SuperPET Waterloo 6809 Assembler SuperPET Waterloo MicroBASIC SuperPET Waterloo MicroFORTRAN SuperPET Waterloo MicroCOBOL SuperPET Waterloo MicroPASCAL SuperPET Waterloo MicroAPL	Boswell/Grove/McPhee/Welch Cowan/Shaw Graham/McPhee Dirkson/Welch Commodore Boswell/Grove/Welch Wilson/Wilkerson	Sam's Books Sam's Books Sam's Books Sam's Books Sam's Books Sam's Books Sam's Books
6502 Programming Books. 6502 Assembly Language Programming 6502 Assembly Language Subroutines 6502 Software Gourmet Guide & Cookbook Advanced 6502 Programming Programming The 6502 6502 Software Design 6502 Assembly Language Programming 6502 Systems Programming Beyond Games: Systems Software for your 6502 Personal Computer Top Down Assembly Language Programmer For Your 6502 Personal Computer Machine Language For Beginners	Lance Leventhal Lance Leventhal Findley Zaks Zaks Scanlon Fernandez/Tabler/Ashley Windeknecht Skiers Skier Mansfield	Osborne Osborne ? Sybex Sybex Sams WILY LB BYTE McGraw Hill Compute!
6809 Programming Books. 6809 Assembly Language Programming Programming The 6809 The 6809 Companion	Lance Leventhal Zaks/Labiak James	Osborne ? BP
Hardware Related Books. PET/CBM & The IEEE 488 BUS (GPIB) Interface Projects For The PET/CBM PET Interfacing Advanced 6502 Interfacing Programming & Interfacing the 6502 with experiments How To Program and Interface the 6800 6809 Microcomputer Programming and Interfacing, with Experiments 6502 Applications	Fisher/Jensen Hallgren Downey/Rogers Holland De Jong Staugaard Staugaard Zaks	Osborne ? Sams Sams Sams Sams Sams Sams Sybex

Title	Author	Publisher
Vic 20 Books. Vic Programmers Reference Guide Compute's First Book Of Vic Compute's Second Book Of Vic Compute's First Book Of Vic Games Users Handbook to the Commodore Vic 20 Vic-20 Computer Writing Workbook 34 Vic-20 Computer Programs for Home, School & Office 101 Programming Tips & Tricks For The Vic-20 Start With BASIC On Your Commodore Vic 20 Vic BASIC: A User-Friendly Guide More Than 32 BASIC Programs for the Commodore Vic Computer Computers for Kids-Commodore Vic-20 Edition Getting Acquainted with your Vic-20 Vic Graphics Vic Revealed Vic Games Mapping The Vic	... Compute! Books Compute! Books Compute! Books Weber Adler Adler Adler Monro Zamora Rugg/Feldman/Wilson Larsen Hartnell Nick Hampshire Nick Hampshire Nick Hampshire Compute!	Sams Compute! Books Compute! Books Compute! Books WSI ARCsoft ARCsoft ARCsoft REST PH Dilithium Press Creative Comp. Press Creative Comp. Press Hayden Hayden Hayden Compute! Books
C64 Books. Commodore 64 Programmers Ref. Guide Compute's First Book of 64 More Than 32 BASIC Programs for the Commodore 64 Computer Mapping The Commodore 64	Commodore Compute! Rugg/Feldman/Moore Compute!	Sams Compute! Books Dilithium Press Compute! Books
Pet/CBM Books. Programming The PET/CBM The PET Revealed The Library Of PET Subroutines PET Graphics CBM Professional Computer Guide PET Personnel Computer Guide Hands-On BASIC With PET Compute's First Book Of PET/CBM PET Basics PET Fun and Games Mostly BASIC: Applications for your PET PET BASIC 1 32 BASIC Programs for the PET Computer PET/CBM An Introduction To BASIC Programming and Applications PET/CBM BASIC PET Fun And Games A PET In The Classroom Instant BASIC	Raeto Collins West Nick Hampshire Nick Hampshire Nick Hampshire Osborne Osborne Peckham Compute! Books David Jeffries and Fisher Berenbon Zamora Rugg Streitmatter/Goldstein Haskell Jeffries Boren Brown	Compute Books Hayden Hayden Hayden Osborne Osborne Osborne Compute! Books Dilithium Press ? Sams PH Dilithium Press BRDY PH Osborne Dilithium Press ?

Glossary

Access Time	- The time required for a computer to locate and transfer data to or from a storage medium.
Acoustic Coupler	- The two rubber cups on an acoustic modem that hold the receiver in place to allow transmission and reception of data.
Acoustic Modem	- A telephone modem that is not hardwired into the telephone lines, but uses a microphone and speaker to transmit data to and from the telephone.
Accumulator	- A location within the microprocessor that is used for storage of the result of an arithmetic or logic operation.
Address	- A location between 0 and 65535 in computer memory.
Algorithm	- A set of rules, usually mathematical, that help in solving a particular problem.
ANSI	- American National Standards Institute.
APL	- Advanced Programming Language.
ASCII	- The 'American Standard Code for Information Interchange'. Is an eight level code serial transmission of alphanumeric code. The first 7 bits represent 128 standard ASCII characters. The eighth bit is a parity bit for error checking. The eight data bits are preceded by one start bit and followed by one or two stop bits.
Assembler	- A computer program that translates source code, ie. the actual written assembler program, into a machine readable form, the object code.
Backup	- The exact duplication of a diskette by the use of a dual disk drive and the backup procedure.
Bank Switching	- A method in which switching between various RAM locations can be achieved through software.
BASIC	- Acronym for Beginners All-purpose Symbolic Instruction Code.
Baud Rate	- The rate of which data is transmitted, which is expressed in terms of bits per second. Baud rate/10 equals bytes per second (approx.)
Binary Number	- A numbering system that uses either the digits 0 or 1 to express all numeric values. The binary digit is also known as the 'bit'.
Bit	- One eighth of a normal byte, in binary it is either a 0 or a 1 symbolizing yes or no, on or off, etc.
Block	- 256 characters of information, as stored on a diskette. Also known as a sector of a diskette.
Buffer	- RAM used for temporary storage of data until the data can be accepted by the device it is being sent to.
Bus	- The lines connecting the memory, microprocessors, and other portions of the computer or peripherals that send or receive data.
Byte	- The equal of one character of information, which is 8 bits. Numeric value of one byte can fall between 0 and 255 decimal.
CPU	- Central Processing Unit, the microprocessor.
COBOL	- COmmon Business Oriented Language.
COMAL	- COmmon Algorithmic Language. Written for Commodore machines in Denmark.
Compiler	- A computer program that will accept a program written in a language such as BASIC, then convert it over to a form that the computer will immediately understand, a machine language.
Daisy Wheel	- A print wheel that uses impact of a solid character to produce its impression. The daisy wheel is circular in design to allow for the wheel to spin around quickly to the correct character.
Density	- The number of bits in a single linear track measured per unit of length of the recording medium, ie. single or double density diskettes.
Data Transmission Rate	- The BAUD rate.
Direct Access	- A method of accessing information from a storage medium, such as a diskette, where the access time is not affected by the location of the data on the disk.

Glossary

Directory	– A list generated by the disk drive of all files currently stored on its diskette.
Diskette	– A thin, flexible magnetic disk permanently enclosed in a semirigid protective jacket.
DOS	– The 'Disk Operating System' which controls all of the actions of the disk drive.
Dot Matrix Printer	– A printer head that forms its characters by the use of numerous metal pins to punch out the image required on paper.
Dynamic memory	– A form of random access memory that maintains its stored information by constant refresh cycles.
Dynamic	– In a constant state of change. (ie. variables)
EPROM	– Erasable Programmable Read Only Memory. Is a form of ROM that can be erased by ultraviolet light, and re-programmed when necessary.
EAROM	– Electrically Alterable Read Only Memory. Is a form of ROM that will remain constant under normal operating conditions, but whose contents can be modified by a series of deliberate electrical signals.
File	– A collection of related records brought together as one, single unit.
File Type	– Pertaining to Commodore DOS, there are four different file types. These are PProGram, SEQuential, RELative, and USEr.
Flag	– A single storage bit used to signal the occurrence of a certain condition.
Floppy Disk	– A diskette
Flow Chart	– A logical diagram of the order in which a particular problem is thought to be resolved. Usually required when complex computer programs are to be written.
Formatting	– Pertaining to Commodore DOS, is the process of organizing a diskette into specific tracks and sectors to allow for storage and retrieval of information.
FORTRAN	– An acronym for FORmula TRANslating system, a high level computer language.
Full Duplex	– In communications, the simultaneous transmission and reception of data, unlike half-duplex.
Graphic Plotter	– A device that connects to the computer, and is controlled by the computers software to draw exact patterns as determined as the computed X and Y co-ordinates.
Half-Duplex	– In communications, allowing only one way data transmission or reception at any one single time. Compares in principal to an inexpensive walkie talkie.
Hard-Copy	– Refers to any printed output in general.
Hard Disk	– A totally self enclosed data storage system. The disk itself is sealed inside the unit, and rotates at approximately 3500 RPM to allow for high speed access to information. A special feature of this type of storage system is the vast amount of storage space available in relation to a floppy disk system.
Hardware	– The actual physical components of a computer system, ie. computer, peripherals, magnetic storage media, paper etc.
Hexadecimal	– A numbering system with the base 16. Its digits range from 0 through F.
High-Level Language	– A computer language such as BASIC, COBOL, FORTRAN, PASCAL, APL and PL/1, to name a few, that use english like commands in the design of computer programs.
Impact printer	– A printer which forms its characters by the physical striking of the ribbon and paper.
Increment	– To increase the value of by a certain quantity.
Interactive language	– A high-level language that allows execution, modification, then re-execution again, for as many cycles as required. BASIC is one example of such a language.
Interface	– A wiring device to connect two or more devices together in a harmonious existence, where data can be transmitted or received accurately.
Interpreter	– A computer program that translates and executes each statement in a program entirely before attempting to continue on to the next statement in line. This is a built in feature of Commodore BASIC.

Glossary

Interrupt Driven	– A program whose execution is activated by the computers operating systems attempts to check its various interrupt routines. Similar to a water wheel placed in a fast moving river. Very little change in the flow if the river but the water wheel is allowed to operate.
I/O	– An abbreviation for Input/Output, or the transmission to and from the microprocessor.
Jump	– In assembler, a branch to a section of code other than what the normal program flow would expect.
K	– In reference to storage capacity, it is equal to 1024 bytes of data.
Light Pen	– A hand-held, light-sensitive device that is wired directly to the computer to allow the light-pen/screen positions to be transmitted to the computer for analysis and re-action.
Low-Level Language	– A computer language, such as assembler, that requires very little translation to become acceptable by the computer.
Machine Language	– A programming language that is immediately understandable by the computer, therefore not requiring any translation at all. In reality not a language as such but a coding of binary digits.
Matrix Printer	– Dot Matrix Printer.
Memory Map	– A listing of addresses and corresponding functions for all RAM and ROM locations within the computers memory.
Microprocessor	– The central processing unit of a microcomputer containing the control unit, arithmetic logic unit, and a number of storage location, all within one small silicon chip.
Modem	– A device that MODulates and DEModulates signals transmitted over the telephone lines, from analog to digital and vice-versa.
Monitor	– A video-display, similar to a television screen, that is used for computers and other electronic devices requiring a stable video output.
Mnemonic Code	– A set of abbreviated commands designed to help make the writing of an assembly language program simpler for the programmer.
Network	– A number of computers and peripherals connected together through either a form of timesharing system, or over the telephone lines in a similar manner.
Nibble	– A group of 4 bits, or one half of a byte.
Null String	– A string variable without any characters in it.
Object Code	– The compiled, or machine code version of what was originally a source code program.
Operand	– The address of a unit of data to be operated upon.
Operating system	– The software that controls the execution of all computer programs, the diagnostic system to help in the debugging process, all input/output, all housekeeping duties of the computer, and all else that go into the running of the system. In Commodore, the operating system is located in ROM.
Output device	– A device, or one in a series of devices used to collect data from another device. To alleviate confusion, the printer is an output device when accepting data from the computer to be printed.
Pascal	– A high-level programming language.
Parallel	– A type of port or interface that allows all eight bits of data to be transmitted at the same time, each through a separate piece of wire.
Peek	– A BASIC keyword used to look at a position in memory to determine the value located there.
Peripheral	– Any number of external devices that can be used with the computer to increase its productivity.
Pixel	– One graphic point on the computers screen. The type of computer, the graphic mode of operation it is in, and the resolution capabilities of the screen determine how high of resolution will be viewed.
Poke	– A BASIC keyword used to store a numeric digit between the values of 0 and 255, within memory.
Port	– A connection point on a computer to connect an external interface to with some form of peripheral.

Glossary

PROM	– Programmable Read Only Memory. A memory storage chip that may be programmed by someone other than the manufacturer. This chip can only be written to once, and becomes a permanent modification when the process is complete.
QWERTY	– On a full sized typewriter keyboard, or computer terminal, the first six letters in the third row are QWERTY
RAM	– Random Access Memory. As the name implies, this memory is such that access to it can be made in a total random sequence. When the power is turned off with this type of memory, all contents are lost.
Read	– The operation performed by the computer as it takes in new information from an input device.
Read/Write Head	– The piece of hardware that can take information from, or write information to a magnetic storage medium.
Register	– A number of small temporary storage locations within the computer, that are used for such things as the operating system, and other functions that the programmer may require.
Resolution	– The number of pixels (points) that can be displayed on a computer screen both horizontally and vertically.
ROM	– Read Only Memory. As the name implies, the contents of this section of memory can be read from, but not written to. When the power is turned off, the contents of ROM are retained.
Sector	– An area on diskette that is comprised of 256 characters of information. Also know as the block, this block is read and write compatible, and can be accessed by the experienced programmer, or left up to the devices of the disk drive itself.
Sequential Access	– A manner in which to read or write data as complete blocks of information, with each byte accessed in the exact order in which was stored.
Serial	– A type of port or interface that allows only one bit of data to be transmitted at a time through one the data line.
Software	– A computer program and its supporting documentation.
Source Code	– A high level language program that requires interpretation or translation into a form acceptable by the computer before processing.
Static Memory	– A form of RAM whose contents are stable as long as the computer is turned on. Unlike dynamic RAM, there is no refresh cycle required.
Storage Capacity	– The amount of data that a device can accept for storage, usually expressed in bytes.
String	– A set of alphanumeric characters stored within memory that can be recalled, modified, or transmitted to peripheral device such as a printer, disk drive, cassette drive, modem, etc.
Syntax	– The correct grammatical design of a computer program according to the particular rules set down when the computer language was designed.
Synchronous	– Occurring simultaneously with a regular or predictable time relationship.
Thermal Printer	– A dot–matrix printer producing its printed characters by the use of a dot–matrix pattern that has been adapted electrically and heat sensitive paper.
Track	– With reference to a diskette, a track is one circular portion of a diskette surface that is further subdivided into sectors, or blocks of information.
Variable	– An area of identifiable storage space that can be read or altered as necessary. Variable length cannot exceed 255 characters.
Volatile Memory	– A form of computer memory that loses its contents after the power has been turned off. Random access memory, or RAM is an example of this.
Write	– A store a section of computer memory, or a quantity of produced data on an external storage medium such as diskette or cassette.
Write Protect	– In order to avoid losing stored data on cassette or diskette storage mediums, write protecting is a method to stop any unintentional damage.



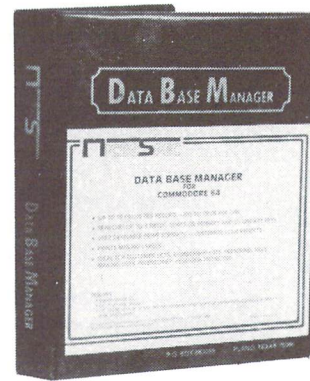
MICROSPEC

SOFTWARE MEANS BUSINESS FOR THE COMMODORE 64

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

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

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



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

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



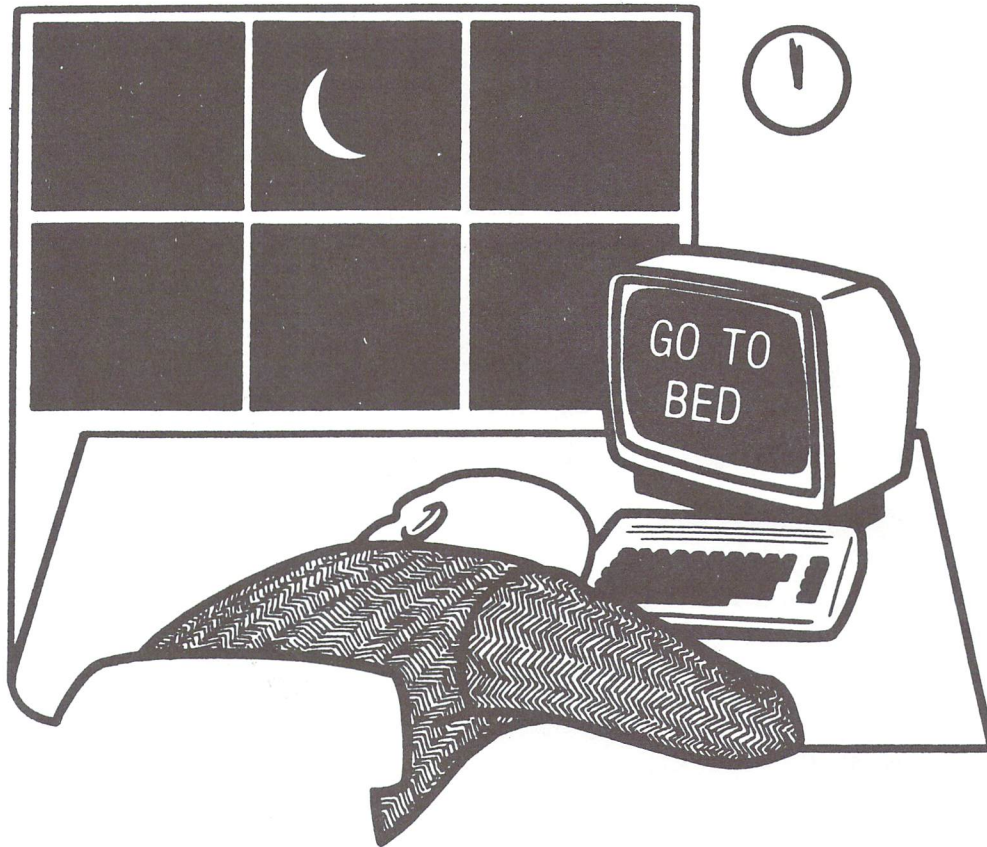
P.O. BOX 863085 • PLANO, TX 75086
(214) 867-1333

The
MIDNITE
SOFTWARE GAZETTE

www.Commodore.ca
May Not Reprint Without Permission

The
PAPER

Five years of service to the PET community.



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

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

635 MAPLE □ MT. ZION, IL 62549 USA

NEW

NEW

MPI INTRODUCES SUPER ACTION MEMORY EXPANDER BOARD FOR VIC 20*

Adds 24K and 3 Expansion Slots With Switches and Fuse \$109.95
(Expandable To 35K By Simply Adding Memory Chips and Switches)

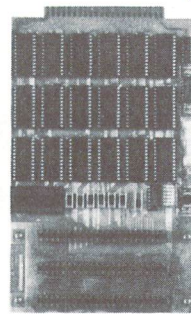
INTRODUCTORY PRICE

\$109.95

24K BOARD FEATURES:

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

Transfer Cartridges To Tape
Evade Self Destruct Code With Write Protect Switch
All Boards Factory Tested
Complete Instructions And Illustrations Included.
Fuse Protected
Reset Switch



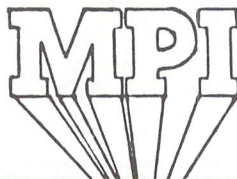
Pictured Above—
Action Board with all options

- 24K memory, 3 expansion slots, switches assembled and tested \$109.95
- Same as above with sockets that allow you to later add your own memory chips to bring memory from 24K to 35K \$124.95
- Full 35K memory, 3 expansion slots, 3K expander mode, eeprom socket (switch selectable between BLK 3 & BLK 5) and all switches assembled and tested (eeprom not included) \$149.95
- Bare 35K board with complete instruction and parts list \$39.95

Prices are in US dollars.

COD Orders: 816-444-4651

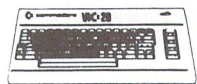
MIDWEST PERIPHERAL INDUSTRIES



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

MPI
Box 8123-B
Kansas City, MO 64112

Personal checks—Allow 3 weeks to clear.



VIC 20* is a registered trademark of Commodore.

NEW

NEW



Reduction of an actual sign

The Banner Machine™

For the Commodore 64 (with 5 fonts) or the VIC-20 with 24K memory. • Use on any Epson MX with Graftrax or the FX and RX printers. • Menu-driven program operates like a word processor. • Makes signs up to 10" tall by any length. • Makes borders of variable width up to ¾". • 8 sizes of letters from ¼" to 6½" high. • Proportional spacing; Automatic centering; Right and left justifying. \$49.95 Tape or Disk (Specify computer equipment)

For the Commodore 64:

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

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

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

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

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

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

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

For the VIC-20:

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

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

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

Cardinal Software™

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

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

Prices are in US dollars.

NEW

Assembler for the
Commodore 64

PAL 64

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

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

\$99.95 from your local Commodore dealer.

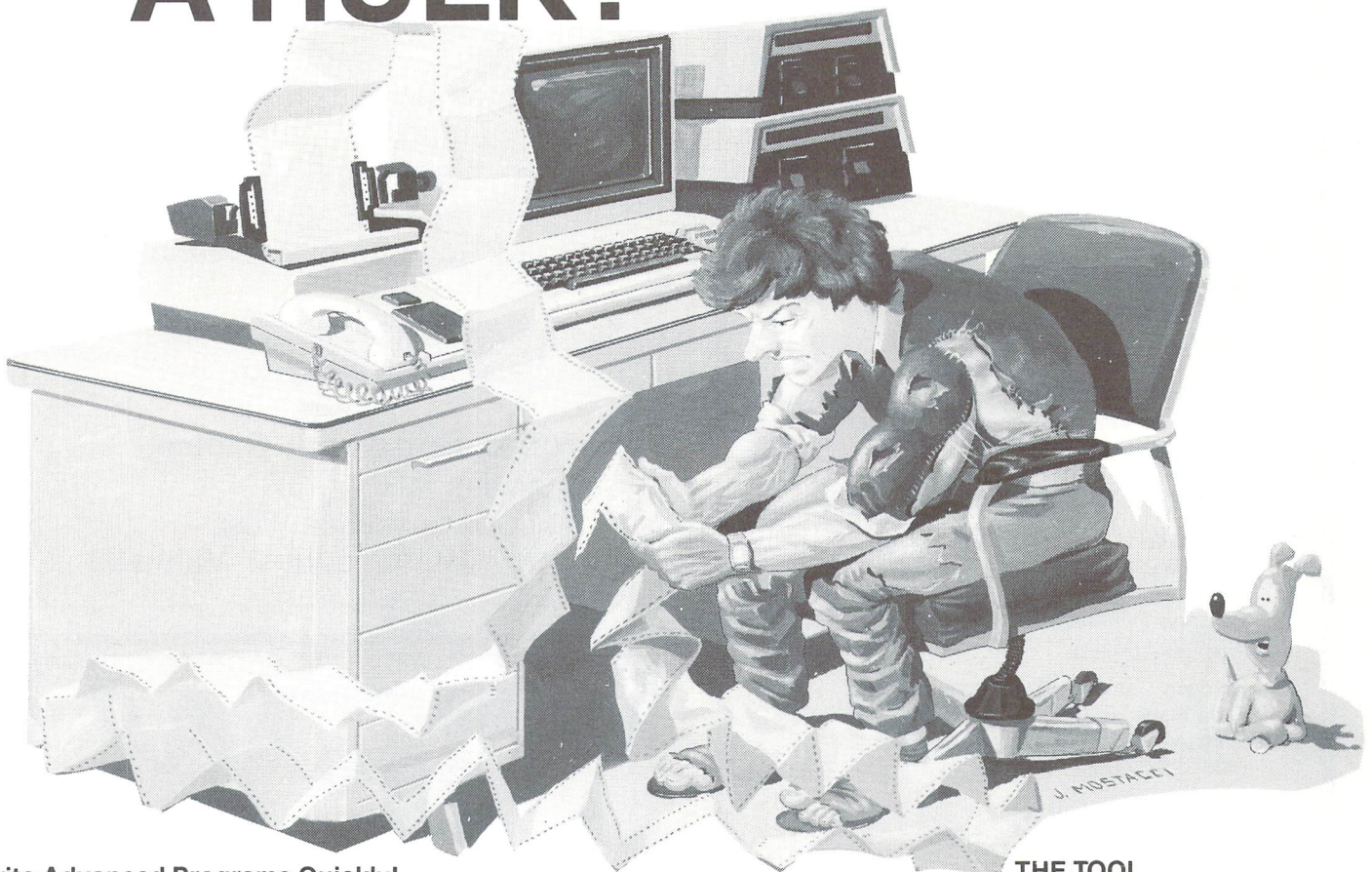
For your nearest dealer call:

(416) 273-6350

PRO-LINE
SOFTWARE

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

IS PROGRAMMING TURNING YOU INTO A HULK?



Write Advanced Programs Quickly!

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

Features of **THE TOOL** include :

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

THE TOOL
For The Commodore 64™
\$65.00

developed by Micro Application

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

Dealer Inquiries Invited

Name _____

Address _____

Prov/State _____ Postal/Zip Code _____

Order VISA MasterCard Cheque

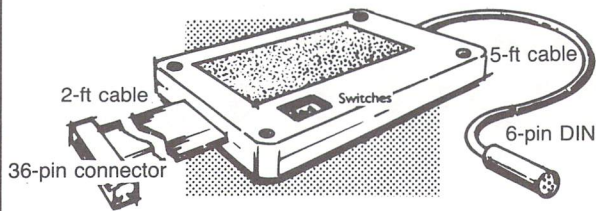
Acc# _____ Expiry _____

Please include numbers above name

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

MW-302: VIC-20/64

Parallel Printer Interface.



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

MW-302 Canadian \$189.95

Micro World Electronix, Inc.

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

CANADIAN DEALERS

ALBERTA

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

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

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

BRITISH COLUMBIA

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

ONTARIO

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

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

SASKATCHEWAN

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

COMMODORE USERS

Join the largest, active Commodore users group in North America and get—

- Access to club library of over 3000 free programs.
- Informative club newsletter.
- The latest information about the PET, CBM, VIC, Super-PET and Commodore-64.

Send \$20.00 (\$30.00 overseas) for Associate Membership to:

Toronto Pet Users Group

Department "M"

381 Lawrence Avenue West

Toronto, Ontario, Canada M5M 1B9

NEW

Basic Utility for the Commodore 64

POWER 64

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

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

\$99.95 from your local Commodore dealer.

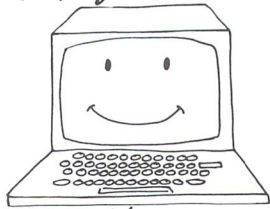
For your nearest dealer call:

(416) 273-6350

PRO-LINE
SOFTWARE

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

(M)agreeable™



software

STOCK HELPER™
 Commodore 64 and VIC-20

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

Stock HELPER is available on diskette for:

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

plus \$1.25 shipping (\$1.55 Canadian)

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

(M)agreeable software, inc.

5925 Magnolia Lane • Plymouth, MN 55442
 (612) 559-1108

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

Intelligent Software
 for the Commodore computers

Catalog 5/1/83

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

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

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

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

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

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

William Robbins, Box 3745, San Rafael, CA 94912

Disk Software for the Commodore 64™

JOT-A-WORD™

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

ONLY \$29⁹⁵

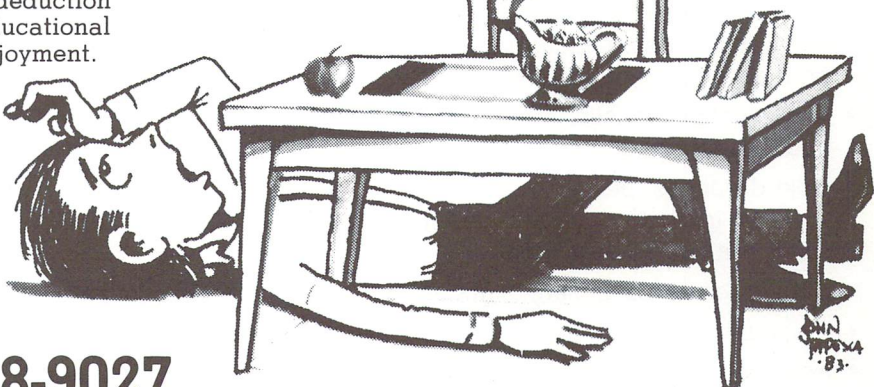
**MICRO
 WARE**

1342B RT. 23
 BUTLER, N.J. 07405

**Dealers & Distributors
 Inquiries Invited**

201-838-9027

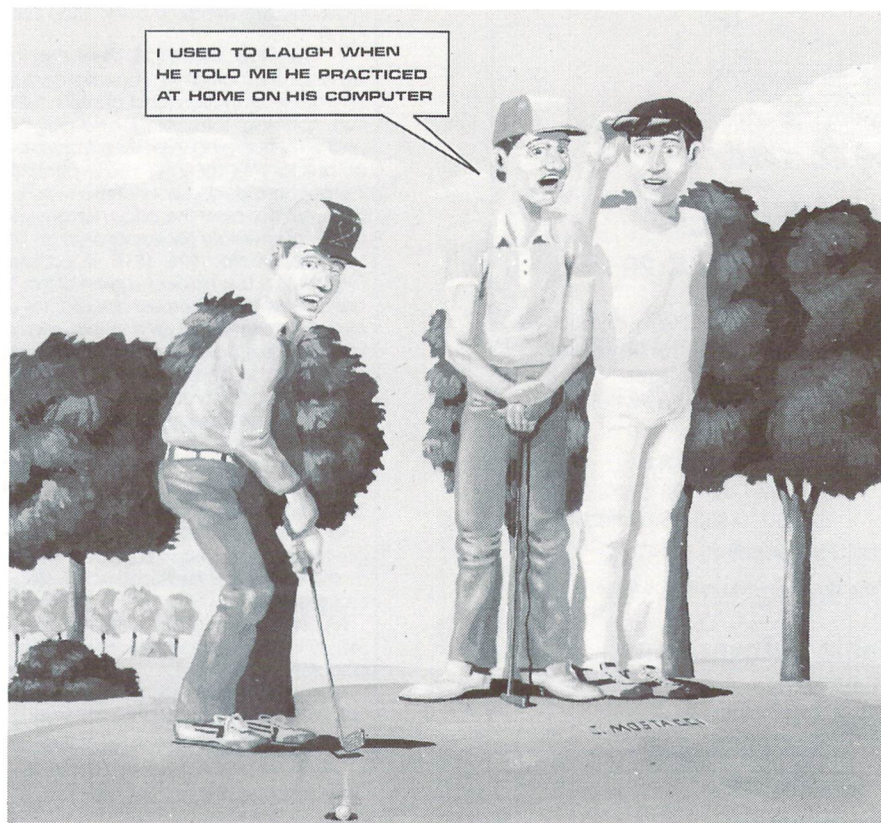
Prices are in US dollars.



"The Genie is hard to beat!"

PRO GOLF

www.Commodore.ca
May Not Reprint Without Permission



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

PRO GOLF Features :

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

PRO GOLF
For The Commodore 64
\$34.95

written by George Adams

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

Dealer Inquiries Invited

Name _____

Address _____

Prov/State _____ Postal/Zip Code _____

Money Order VISA MasterCard Cheque

Acc# _____ Expiry _____

Please include numbers above name

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

Let The SMART 64 Terminal

COMMODORE 64*

Do The DRIVING

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

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

So why not travel the communications highways the SMART way!

Accessories included:

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



Suggested
\$39.95
Retail

*Commodore 64 registered trademark of Commodore Business Machines Inc.

*Supports 80-column cartridge by Data 20 Corporation.

Dealer Availability
Call (203) 389-8383



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

Prices are in US dollars.

VIC 20™
COMMODORE 64™

JOIN THE COMPUTER REVOLUTION WITH A MASTERY OF THE KEYBOARD!

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

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

TYPING TUTOR PLUS WORD INVADERS

The proven way to learn touch typing.

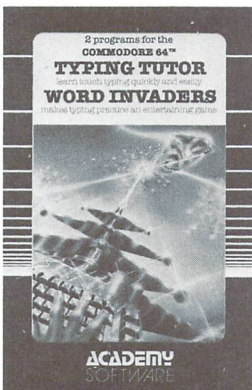
COMMODORE 64 Tape \$21.95 COMMODORE 64 Disk \$24.95
VIC 20 (unexpanded) Tape \$21.95

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

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

Customer comment says it all . . .

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



NEW!

IFR
(FLIGHT
SIMULATOR)
CARTRIDGE
FOR THE VIC 20
\$39.95
JOYSTICK REQUIRED



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



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



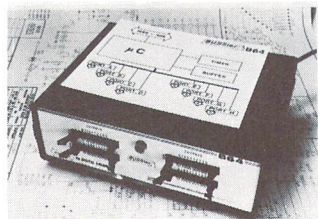
ACADEMY
SOFTWARE

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

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



REAL-WORLD INTERFACES FOR COMPUTERS



ANALOG AND DIGITAL INPUT/OUTPUT

The BUSSter line of analog and digital products was designed to collect data and to output signals to laboratory and industrial equipment in conjunction with a microcomputer system. These powerful self-contained modules reduce a computer's workload by providing read or write operations to external devices. They are controlled as slave interfaces to real-world physical applications. Control is over an IEEE-488 (GPIB) bus or RS-232 port. The internal buffer and timer provide flexibility by allowing the BUSSter to collect data while the host computer is busy with other tasks. All units have a timer which operates from .01 seconds to 46 hours.

- **BUSSter** — 64 channel digital input module to read 64 digital signals. Built-in buffer \$495.00
- **BUSSter B64** — 64 channel digital output module to send 64 digital signals . . . \$495.00
- **BUSSter C64** — 64 channel digital input/output module to read 32 and write 32 digital signals. Built-in buffer \$495.00
- **BUSSter D16** — 16 channel analog input module to read up to 16 analog signals with 8 bit resolution (1/4%). Built-in buffer . . . \$495.00
- **BUSSter D32** — 32 channel version of the D16. \$595.00
- **BUSSter E4** — 4 channel analog output module to send 4 analog signals with 12 bit resolution (.06%) \$495.00
- **BUSSter E8** — 8 channel version of the E4 \$595.00
- **BUSSter E16** — 16 channel version of the E4 \$695.00

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

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

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

US Dollars Quoted—\$10 Shipping & Handling
 MASTERCARD / VISA



Connecticut microComputer, Inc.
INSTRUMENT DIVISION

36 Del Mar Drive • Brookfield, Ct. 06804
 (203) 775-4595 TWX: 710-456-0052

CURSOR

For your Commodore 64

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

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

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

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

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

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

COMPARE THESE FEATURES:

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

MAILPRO 64.....\$129⁹⁵

Also available for **COMMODORE 8032 . . . \$179⁹⁵**

Call for the name of your local dealer:

PRO-LINE
 SOFTWARE
 PRO-LINE SOFTWARE LTD.

(416) 273-6350

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

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

For your **VIC-20®**, **C-64®**
ATARI®

Centronics Parallel Types
 And RS-232 Serial Types

SUG.
 LIST
 \$299

\$229.95!!

BASIC PRINTER
 (Requires one
 Option Below)

FEATURES:

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



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

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

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

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

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

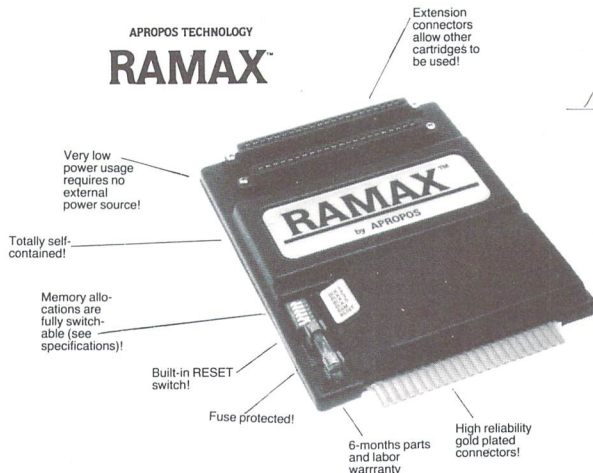
ADD: \$8.00 shipping (cont. USA), \$25.00 (Canada, HI, AK)

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

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

Maximum Memory allows you to use more powerful programs for:

- EDUCATION • ENTERTAINMENT • MAIL LISTS
- BUSINESS APPLICATIONS • FINANCIAL RECORDS



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

SPECIAL LOW PRICE!

Only \$149.00

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

10 DAY MONEY-BACK GUARANTEE

If not satisfied, simply return in original condition for your money back.

RAMAX Jr.™

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

Special Only \$129.00
Shipping included

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

RAMAX™ Features and Specifications:

- Adds up to a full 27k bytes of additional RAM to the standard VIC-20's internal RAM of 5k.
- Built-in switch allows User selection of any combination of 5 areas or RAM memory*:
 BLK1 (8k: Adr. 8192-16383)
 BLK 2 (8k: Adr. 16384-24575)
 BLK 3 (8k: Adr. 24576-32767)
 BLK 5 (Adr. 40960-49151, allows/disallows 8k ROM games)
 RAM (3k: Adr. 1024-4095)
 RESET (Resets computer without power off/on)
- Built-in electrical Fuse to protect equipment.
- Totally self-contained. No external power supply needed.
- Two (2) extension connectors allow ANY additional cartridges and/or devices designed for the VIC expansion port.
- Very low power consumption (.175 amp usual).
- High reliability gold-plated connectors are designed for long life.
- Complete Operating Manual.
- 6 month parts and labor warranty to original purchaser.
- Factory service.

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

New Product!

APROSPAND-64™

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

Shipping included

TO ORDER:

Send Check or Money Order For the Total
 Calif. residents add 6% tax.
 Or Contact your Local Dealer

Phone orders Call **(805) 482-3604**



All Prices U.S. Dollars

DEALER INQUIRIES WELCOME

WE SERVICE WHAT WE SELL

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

APROPOS TECHNOLOGY
 1071-A Avenida Acaso
 Camarillo, CA 93010

MICROCOMPUTER SUPPLIES



100%
Guaranteed

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

Shop
Without
Going
Shopping

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

All other sizes and format available and volume discount.

Flip 'N' File, hold 50, 5 1/4" diskettes	\$38.50 each
Flip Pac, holds 10 5 1/4" diskettes	\$ 7.00 each
Disk Banks, hold 10 5 1/4" diskettes	\$ 7.00
Plastic Library Cases, choice of colors;	\$4.50 each
EPSON RX80 Printer	\$525.00
EPSON FX80 Printer	\$850.00
EPSON FX100	\$CALL
QUADRAM Interfaces	\$CALL
COMREX CR11 Daisy Wheel Printer;	\$869.00
COMREX Monitors, 8 models	from \$198.00
MX80 Ribbons	\$14.50

¹ WRITE OR CALL FOR FREE CATALOG

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

INTERNATIONAL MARKETING SERVICES

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

FREE CATALOG!

UTILITIES, GAMES, AMATEUR RADIO
FOR THE VIC 20 AND COMMODORE 64

NEW ITEM...

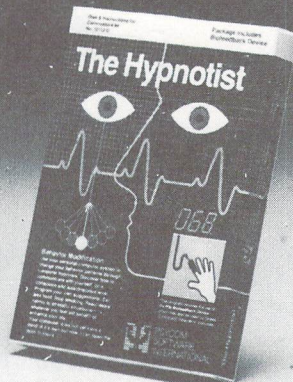
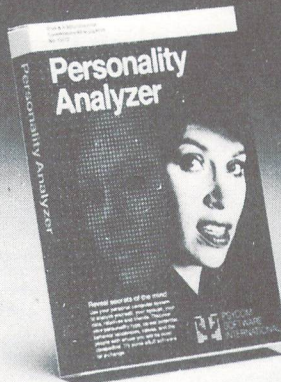
C64 WORLD CLOCK - graphic map showing time in cities across the world.....
Tape \$ 7.95
Disk \$10.95
(+ \$2.00 shipping - US. Currency)

Over 30 other programs!

Low prices! Excellent service!

RAK Electronics
PO. Box 1585
Orange Park, FL
32067-1585

Open your mind



© 1983 PATRICK J. WILLIAMS

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

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

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



**PSYCOM
SOFTWARE
INTERNATIONAL**

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

**This
Space
Could
Be
Transacting
For
You!**

Kelly M. George
Advertising Manager
416 876 4741

TELSTAR 64

Sophisticated Terminal Communications Cartridge for the 64.

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

Don't settle for less than the best!

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

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

Machine Language Monitor Cartridge for the CBM 64

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

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

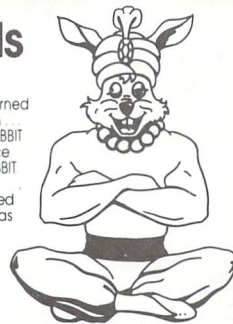
Cartridge and Manual — **\$24.95**

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

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

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

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



\$39.95

MAE NOW THE BEST FOR LESS!

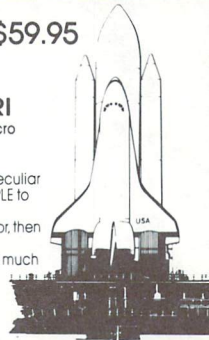
\$59.95

For CBM 64, PET, APPLE, and ATARI

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

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

Still not convinced, send for our free spec sheet!



Eastern House

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



Prices are in US dollars.

C 64 PROVINCIAL PAYROLL

A complete Canadian Payroll System for Small Business.

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

Available from your Commodore Dealer.

Distributed by:



**MICROCOMPUTER
SOLUTIONS**

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

TYPRO DATA MANAGER & WORD PROCESSOR

For COMMODORE 8032 Computer — 8050/4040 Dr.

NOW AVAILABLE FOR THE COMMODORE 64®

DATA MANAGER

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

WORD PROCESSOR

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

BOTH PROGRAMS ABOVE, ONLY \$89.00

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

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

All software is fully supported for updates and revisions for up to six months after purchase.

Specify Computer model number and Disk model number.

INPUT SYSTEMS, INC.

25101 S.W. 194 Ave. Homestead, FL 33031 (305) 245-3141

DEALER INQUIRIES INVITED.

VIC-20

GOSUB

www.Commodore.ca
May Not Reprint Without Permission

C64

INTERNATIONAL INCORPORATED

The Flexikey System



Retail
\$69.95

Features:

19 Keys, each of which may have 3 separate definitions!

Complete documentation including program listings!

Works on the VIC20 (Expanded) and C-64 computers!

Compatible with most existing software!

Great for use with business programs and electronic spread sheets!

Ideal for machine language programmer!

VISA & MASTERCARD WELCOME

Prices subject to change

Dealer Inquires Invited - (316) 265-9858
GOSUB International - 501 E. Pawnee - Suite 430
Wichita, Kansas 67211

*C-64 and VIC 20 are registered trademarks of Commodore International.

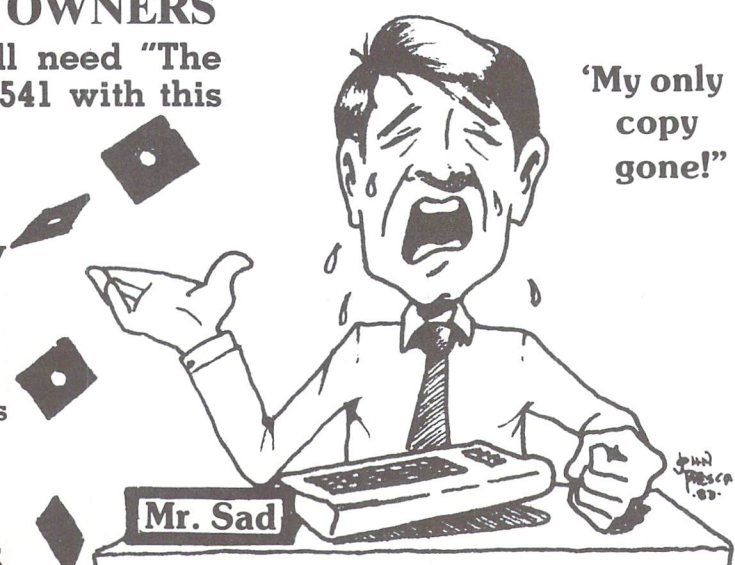
We'll back you up!



ATTENTION COMMODORE 64 OWNERS

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

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



'My only copy gone!'

~~List \$49.95~~

Special intro \$39.95

Dealers & Distributors
Inquiries Invited

CALL (201) 838-9027

MICROWARE

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

Prices are in US dollars.

Commodore 64
and
VIC-20

SuperTerm

\$99⁹⁵



Telecommunications with a difference!

Unexcelled communications power and compatibility, especially for professionals and serious computer users. Look us over; **SuperTerm** isn't just "another" terminal program. Like our famous Terminal-40, **it's the one others will be judged by.**

- **UP/DOWNLOAD FORMATS**—CBM, Xon-Xoff, ACK-NAK, etc.
- **DISPLAY MODES**—40 column; 80/132 with side-scrolling
- **EMULATION**—42 popular terminal protocols
- **FUNCTION KEYS**—8 standard, 52 user-defined
- **BUFFERS**—Receive, Transmit, Program, and Screen
- **EDITING**—Full-screen editing of Receive buffer
- **FILE CONVERSION**—ASCII to PGM, PGM to ASCII
- **PRINTING**—Continuous printing with Smart ASCII and parallel printer; buffer printing with other interfaces or VIC printer
- **DISK SUPPORT**—Directory, Copy, Rename, Scratch, etc.
- **FLEXIBILITY**—Select baud, duplex, parity, stopbits, etc.

Program options are selected by menus and function keys. For maximum convenience, an EXEC file sets options on start-up. SuperTerm may be backed-up for safety. Software on disk or cassette, with special cartridge module.

Write for the full story on SuperTerm; or, if you already want that difference, order today!

Requires: Commodore 64 or VIC-20, disk drive or Datasette, and compatible modem. VIC version requires 16K memory expansion. Please specify VIC or 64 when ordering.

Just need UP/DOWNLOAD?

If you don't yet need SuperTerm's power, perhaps **Terminal-40 Plus** (VIC) or **'64 Terminal Plus** is right for you. We took our top-rated, smooth-scrolling terminal programs, added up/download, disk commands, and even more convenience. Then we put them on disk for fast loading, just like you wanted. Need we say more?

Only \$49.95 (VIC version requires 8K mem exp)

P.S. Trade in your original Terminal-40 or '64 Terminal and deduct \$10.00.

VIC 20 and Commodore 64 are trademarks of Commodore Electronics, Ltd.

(816) 333-7200

Send for a free brochure.



**MIDWEST
MICRO inc.**

MAIL ORDER: Add \$1.50 shipping and handling (\$3.50 for C.O.D.); VISA/Mastercard accepted (card# and exp. date). MO residents add 5.625% sales tax. Foreign orders payable U.S.\$, U.S. Bank ONLY; add \$5 shp/hndlq.

311 WEST 72nd ST. • KANSAS CITY • MO • 64114

Prices are in US dollars.

This
Space
Could
Be
Transacting
For
You!

PRO·LINE SOFTWARE

A CANADIAN COMPANY

designing,
developing,
manufacturing,
publishing
and
distributing
microcomputer
software

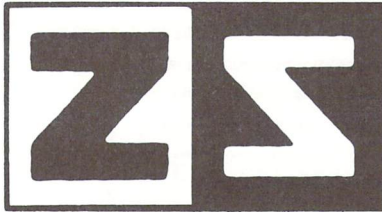
DEALER ENQUIRIES WELCOME
AUTHOR'S SUBMISSIONS INVITED

CALL OR WRITE

(416) 273-6350

**PRO·LINE
SOFTWARE**

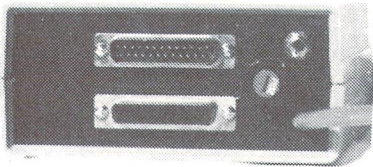
755 THE QUEENSWAY EAST, UNIT 8,
MISSISSAUGA, ONTARIO L4Y 4C5



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

ZANIM SYSTEMS

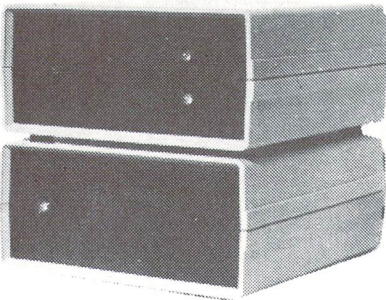
HOME CONTROL FOR YOUR VIC UNDER \$200



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

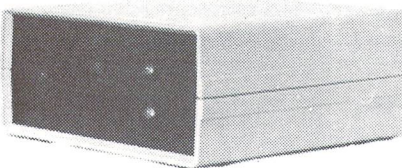
*The ZCM-1V is available for VIC-20 and C-64 users.

VIC/C64 \$129.95
 RS-232 \$149.95



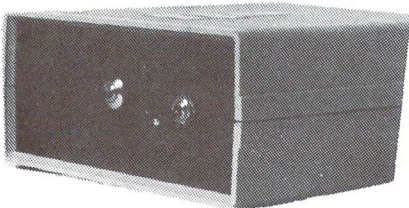
ZAM-1 is the home control interface module that provides a fully versatile computer controlled environment. ZAM-1 can control up to 256 different lamps and appliances in your home or business providing you with an effective and easy to implement energy management and electric control system. No special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

*The ZCM-1/ZCM-1V Master Control module is required to use the ZAM-1 Home Control module.
 \$169.95



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

*The ZCM-1/ZCM-1V Master Control module is required to use the ZAM-2 Security module.
 \$179.95



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

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

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

Prices are in US dollars.

SERIAL OR PARALLEL (CENTRONICS) PORT SWITCHER



**DOES YOUR COMPUTER
 LOOK LIKE THIS?**

**A PORT SWITCHER NOW
 AVAILABLE FOR YOUR
 COMPUTER (ZSW1)**



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

Please send me more information or catalogue!

Name _____

Company _____

Address _____

City _____ State _____ Zip _____

What Make/Model Computer do you own?

SOFTWARE

Commercial Applications For Small Business Computers

- General Ledger
- Accounts Receivable
- Inventory
- Job Costing
- Payroll
- Property Management
- Micrograph
- Law Office Acct.

BEACON SOFTWARE INC.

705 Progress Avenue, Unit 17,
Scarborough, Ontario M1H 2X1
(416) 431-3200

Formerly BPI Micro Systems Ltd.

ASSETS	BALANCE	
CASH ON HAND	268.15	
BANK	(2,633.77)	
ACCOUNTS RECEIVABLE	29,490.58	
INVENTORY	224,012.81	
PREPAID EXPENSE	4,493.65	
TOTAL CURRENT ASSETS		255,630.42
FIXED ASSETS		
LAND	38,083.93	
BUILDING	30,755.11	
FURNITURE & FIXTURES	3,139.56	
VEHICLE	2,137.10	
TOTAL FIXED ASSETS		74,115.70
OTHER ASSETS		
GOODWILL	1.00	
TOTAL ASSETS		29,748.12
LIABILITIES		
CURRENT LIABILITIES		
BANK LOAN		
TERM LOAN		
ACCOUNTS PAYABLE		
TOTAL CURRENT LIABILITIES		
LONG TERM LIABILITIES		
MORTGAGES		
DUE TO SHAREHOLDERS		
LOAN PAYABLE		
TOTAL LONG TERM LIABILITIES		
TOTAL LIABILITIES		10,089.00
CAPITAL		
SHARE CAPITAL		100.00
RETAINED EARNINGS		
TOTAL CAPITAL		66,137.06
TOTAL LIABILITIES & CAPITAL		319,558.41

Featuring: MICROGRAPH

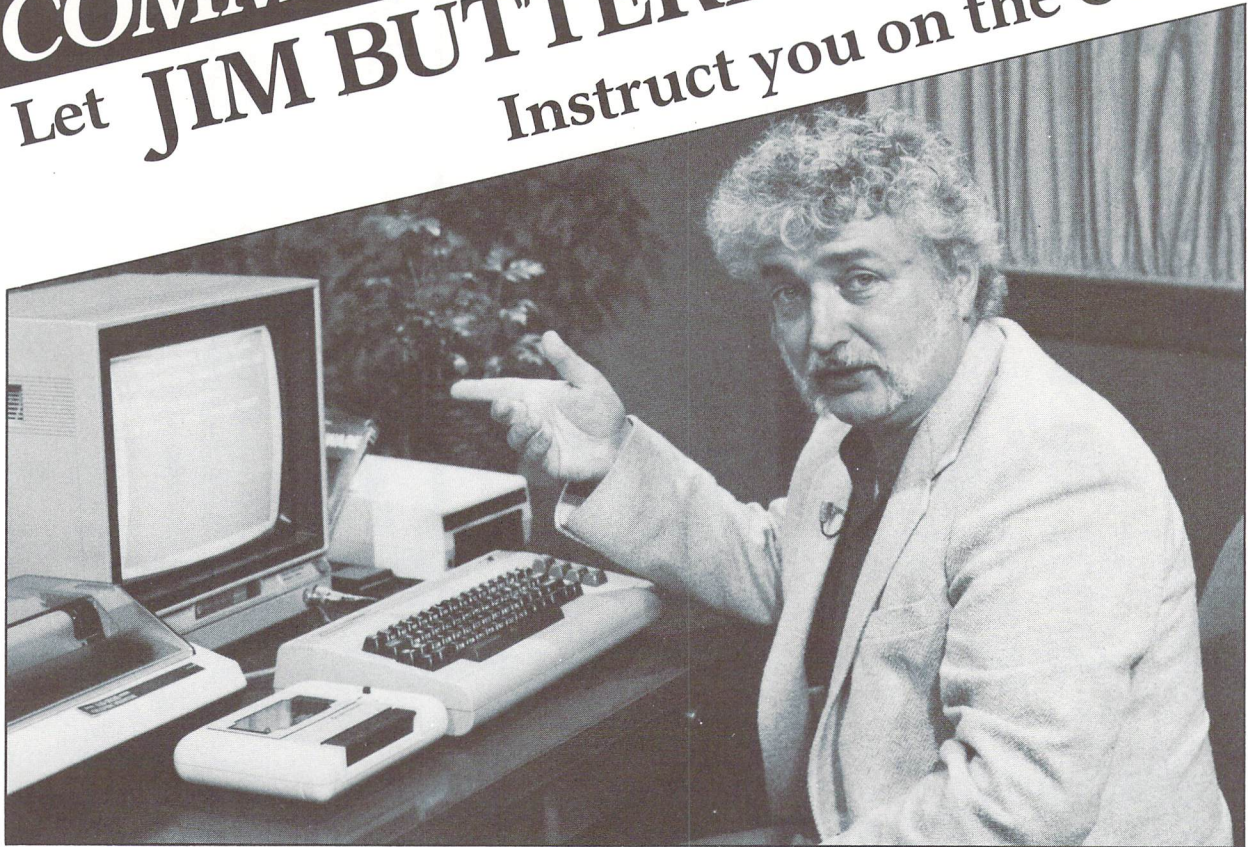
The Micrograph systems combines flexibility, accuracy and easy data entry, to provide the most comprehensive business graphic packages available.

NEW for the COMMODORE 64

Let JIM BUTTERFIELD

Associate Editor
Compute Magazine

Instruct you on the C64



14 SESSIONS ON VIDEO TAPE

- | | |
|---|------------------------------------|
| 1) What Is A Commodore 64? | 7) Computers Talking to Computers |
| 2) Getting Started | 8) Commodore 64 Language |
| 3) Lets Run Programs | 9) Graphics |
| 4-A) What Makes Programs Work? | 10) Commodore 64 Working For You |
| 4-B) Putting Programs To Work | 11) Commodore 64 Music |
| 5) Storing Information | 12) Computer Games And Simulations |
| 6) The Commodore 64
As A Learning Tool | 13) Now What? |

(BETA OR V.H.S.)

Order by phone with **VISA** or **MASTER CHARGE**
(209) 255-1600

Send \$39.95

California residents add
6% sales tax

TO: COMM 64 Training Tape
2727 N. Grove Ind. Drive #101
Fresno, California 93727
Order by Dec. 10 for
Christmas delivery

Look at these Features

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



Wordprocessor for Commodore 64

BLIZTEXT is a trademark of ELCOMP PUBLISHING, INC.

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

Dealer and Distributor inquiries are invited.

BLIZTEXT -- SUPER WORDPROCESSOR for the Commodore-64

— ON SALE NOW! —

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

AVAILABLE NOW!

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

MACROFIRE Editor/Assembler for the Commodore-64

ON SALE NOW
 AVAILABLE IMMEDIATELY

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

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

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

Manual only \$19.95
 Order # 4963 \$89.00

THE GREAT BOOK OF GAMES, VOL. I,

by Franz Ende

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

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

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

NEW PRODUCTS

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

How to program in 6502 Machine Language on your C-64, by S. Roberts (Introduction)
 Order # 184 \$12.95

Commodore-64 Tune-up, Vol. I, by S. Roberts
 How to expand and customize your C-64.
 Order # 185 \$12.95

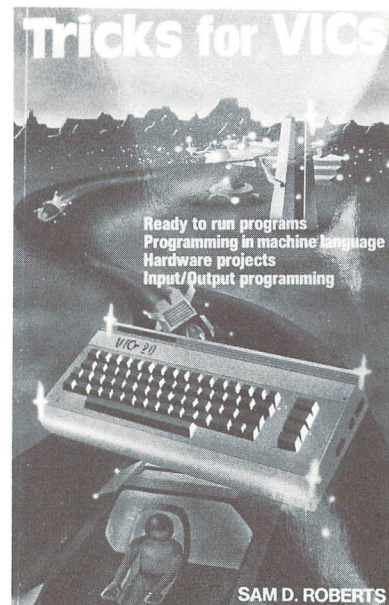
Small Business Programs for the Commodore-64 by S. Roberts
 How to make money using your C-64. Mailing list, invoice writing, inventory, simple wordprocessing and much more.
 Order # 186 \$12.95

Hardware Add-Ons:

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

For your VIC-20

Tricks for VICs \$ 9.95
 Universal Experimenter board



HOFACKER

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

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

Advertising Index

Software

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

Hardware

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

Accessories

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



www.Commodore.ca
May Not Reprint Without Permission

Laser Strike

for the commodore 64



SKENE

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

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

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

Visa/MC/Check/Money Order accepted

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

* Ask about Laser strike posters

Call Toll Free - 1-800-558-8803



In U.K.
Cassette £ 9.00 VAT included
Disk £19.95 VAT included
Isis Hathor U.K.
Andrew Barrow
Royden, Perkslane
Prestwood, Gt. Missenden
Bucks, England HP16 0JD
02406-3224
You will be billed
for postage and handling

COMPATIBLE COLOR-I . . .

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



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

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

Just write, or call to receive complete specifications on the Amdek Color-I Monitor.

- Quality 260(H) x 300(V) line resolution.
- Built-in speaker and audio amplifier.
- Front mounted controls for easy adjustment.
- Interface cables available for Atari and VIC Comodore computers.
- FCC/UL approved.

2201 Lively Blvd. • Elk Grove Village, IL 60007
(312) 364-1180 TLX: 25-4786

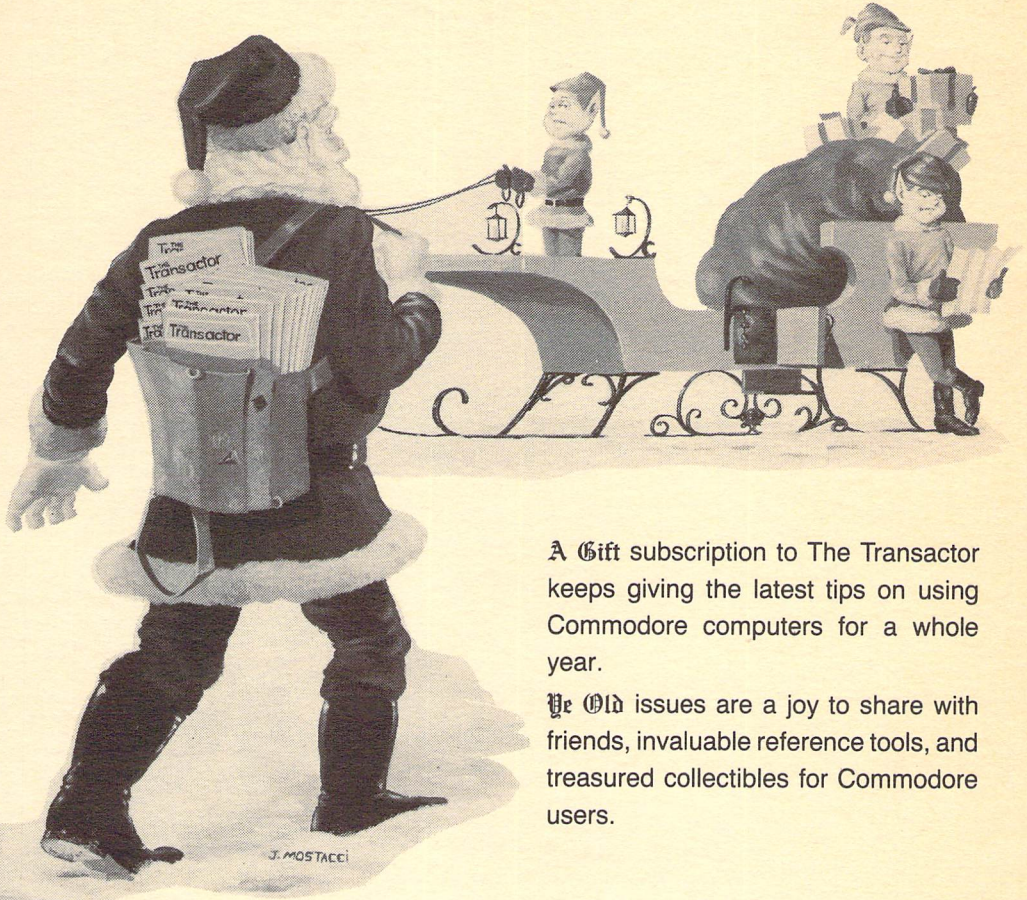
REGIONAL OFFICES: Calif. (714) 662-3949 • Texas (817) 498-2334

AMDEK CORP.

Amdek . . . your guide to innovative computing!



GIVE A GIFT



A Gift subscription to The Transactor keeps giving the latest tips on using Commodore computers for a whole year.

The Old issues are a joy to share with friends, invaluable reference tools, and treasured collectibles for Commodore users.

Season's Greetings from The Transactor

```

100 poke59468,12 : print chr$(142)
110 s$ = "tsiunzsnf!jbsdt"
120 print chr$(147) : p = 193 : d = 215
130 for j = 0 to 15 : k% = j/4 : k = j-2*k% : if k = 0 then k = 1
140 for i = 1 to 19-k : print chr$(32) : nexti
150 k% = k : c = 205
160 k% = k%-1 : if k% > 0 then gosub 260 : goto 160
170 print chr$(p) : p = 214 + rnd(1)*1.1 : c = 206
180 k = k-1 : if k > 0 then gosub 260 : goto 180

```

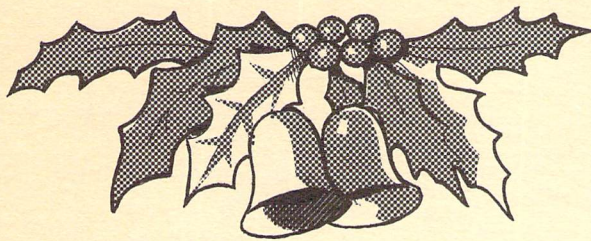
```

190 print : nextj
200 print tab(18) ; "x" chr$(13)tab(17) "xxx"
210 print : print tab(11) ;
220 for j = 1 to 15
230 k = j*4 : k% = k/15 : i = k-k%*15 + 1
240 print chr$(asc(mid$(s$,i))-1) ;
250 nextj : print chr$(19) : p = 193 : goto 130
260 if rnd(1) < .1 then print chr$(d) : return
270 d = 424-d : print chr$(c) : return

```



THAT LASTS



The Transactor
The Tech/News Journal For Commodore Computers

USA

Also send me:
(available for a limited time only,
please send separate cheques for each)

Please send me **6** consecutive **Transactors** starting with the next issue!

Canada \$15. U.S.A. \$15. Foreign \$18. Air Mail (Overseas only) \$25.

Also, please send me the following back issues:
(Add \$2.50 p&h.) Volume 4, Issue #: 1 2 3 4 5 6

The Best of The Transactor Volume 2 (1980/81)	Can. <input type="checkbox"/> \$15.	U.S.A. <input type="checkbox"/> \$18 US.	Foreign <input type="checkbox"/> \$20 US.
The Best of The Transactor Volume 3 (1981/82)	<input type="checkbox"/> \$15.	<input type="checkbox"/> \$18 US.	<input type="checkbox"/> \$20 US.

Name & Address _____

(please include your postal/zip code): _____

NOTE: Prepayment required. Purchase orders will be accepted ONLY if accompanied by payment.

Cheque/MO. enclosed Cheque# _____ Dated ____/____/____ Amount _____

Visa MasterCard Acct. # _____ Expires ____/____

I use the following Commodore equipment:

<input type="checkbox"/> VIC 20	<input type="checkbox"/> C 64	<input type="checkbox"/> 4016/32	<input type="checkbox"/> 8032/96	<input type="checkbox"/> SuperPET	<input type="checkbox"/> 'P' Series	<input type="checkbox"/> 'B'/'BX' Series
<input type="checkbox"/> Datasette	Disk Unit:	<input type="checkbox"/> 1540/41	<input type="checkbox"/> 4040	<input type="checkbox"/> 8050	<input type="checkbox"/> 8250	<input type="checkbox"/> 9060/90

I use my equipment in the following environment:

Hobby Business Technical Public School High School College/Univ. CBM Dealer

Please send me advertising information for The Transactor. Please send dealer information. 01/84

The Transactor
The Tech/News Journal For Commodore Computers

Canada

Also send me:
(available for a limited time only,
please send separate cheques for each)

Please send me **6** consecutive **Transactors** starting with the next issue!

Canada \$15. U.S.A. \$15. Foreign \$18. Air Mail (Overseas only) \$25.

Also, please send me the following back issues:
(Add \$2.50 p&h.) Volume 4, Issue #: 1 2 3 4 5 6

The Best of The Transactor Volume 2 (1980/81)	Can. <input type="checkbox"/> \$15.	U.S.A. <input type="checkbox"/> \$18 US.	Foreign <input type="checkbox"/> \$20 US.
The Best of The Transactor Volume 3 (1981/82)	<input type="checkbox"/> \$15.	<input type="checkbox"/> \$18 US.	<input type="checkbox"/> \$20 US.

Name & Address _____

(please include your postal/zip code): _____

NOTE: Prepayment required. Purchase orders will be accepted ONLY if accompanied by payment.

Cheque/MO. enclosed Cheque# _____ Dated ____/____/____ Amount _____

Visa MasterCard Acct. # _____ Expires ____/____

I use the following Commodore equipment:

<input type="checkbox"/> VIC 20	<input type="checkbox"/> C 64	<input type="checkbox"/> 4016/32	<input type="checkbox"/> 8032/96	<input type="checkbox"/> SuperPET	<input type="checkbox"/> 'P' Series	<input type="checkbox"/> 'B'/'BX' Series
<input type="checkbox"/> Datasette	Disk Unit:	<input type="checkbox"/> 1540/41	<input type="checkbox"/> 4040	<input type="checkbox"/> 8050	<input type="checkbox"/> 8250	<input type="checkbox"/> 9060/90

I use my equipment in the following environment:

Hobby Business Technical Public School High School College/Univ. CBM Dealer

Please send me advertising information for The Transactor. Please send dealer information. 01/84



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

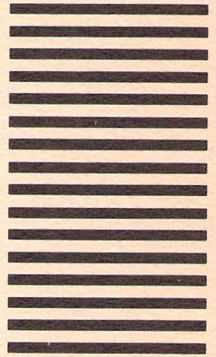
BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 390 BUFFALO, NY

POSTAGE WILL BE PAID BY ADDRESSEE

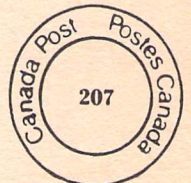
**The
Transactor**

277 Linwood Avenue
Buffalo, NY, 14209



Business Reply Mail
No Postage Stamp Necessary
if mailed in Canada

Postage will be paid by:



**The
Transactor**

500 Steeles Avenue
Milton, Ontario, Canada
L9T 9Z9

