COMMODORE 64

BusCard II®

BATTERIES INCLUDED
TORONTO, ONT. CANADA

B u s C a r d I I

IEEE and Parallel Printer Interface

by Batteries Included

Thank you for purchasing BusCard II.

We believe that BusCard II is the best interface system available for the Commodore 64. BusCard II is easy to use and yet very powerful. BusCard II is probably the only interface you will ever need.

Again, Thank you for choosing BusCard II.

Batteries Included.

Batteries Included
186 Queen Street West
Toronto, Ontario
Canada, M5V-1Z1
(416) 596-1405

Copyright 1984 by Batteries Included

All Rights Reserved

Programming: Steven Douglas
Hardware design: Keith Hope and Jim Law
Produced by Batteries Included

NOTICE OF REGISTERED TRADEMARK

Reference is made in several places in this manual to Commodore and Commodore 64 which are registered trademarks of Commodore Business Machines Inc.

BUSCARD II OWNERS MANUAL

COPYRIGHT 1984 BY BATTERIES INCLUDED

186 Queen Street West
TORONTO, ONTARIO CANADA, M5V 1Z1
(416) 596-1405

This manual and the computer programs contained in or transferred from the BusCard II ROM chip which are described by this manual are copyrighted and contain proprietary information belonging to Batteries Included.

This manual may not be copied, photocopied, reproduced, translated or reduced to machine readable form, in whole or in part, without the prior written consent of Batteries Included.

The ROM chip, and information contained within it, may not be duplicated, in whole or in part, for any purpose. No copies of this manual or the listings of the programs in the ROM chip may be sold or given to any person or other entity.

LIMITATIONS OF WARRANTY AND LIABILITY

Batteries Included, or any dealer or distributor distributing this product, makes NO WARRANTY, EXPRESS OR IMPLIED, with respect to this manual, the BusCard II software and any other related items, their quality, performance, merchantability, or fitness for any particular use. It is solely the purchaser's responsibility to determine their suitability for any particular purpose.

Batteries Included will in no event be held liable for direct, indirect or incidental damages resulting from any defect or omission in this manual, the BusCard II ROM, the BusCard II hardware, the software contained within the ROM chip, or other related items and processes, including but not limited to any interruption of services, loss of business or anticipatory profit, or other consequential damages.

THIS STATEMENT OF LIMITED LIABILITY IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Batteries Included neither assumes nor authorizes any other person to assume for it any other warranty or liability in connection with the sale of its products.

PRODUCT IMPROVEMENTS

Batteries Included reserves the right to make corrections or improvements to this manual and to the related BusCard II device at any time without notice and with no responsibility to provide these changes to purchasers of earlier versions of its products.

Installing The BusCard II	
Unpacking	1-1
Installation	1-2
Setting The Switches	2-1
Device 4 (Printer)	2-1
Devices 5-10 (Including Disk)	2-2
Devices 11 And Up	2-2
General Configuration	2-3
Re-reading The Switches	2-3
Basic 4.0	3-1
Turning Basic 4.0 On And Off	3-1
DS, DS$ And ST	3-2
DOS File Name Pattern Matching	3-3
Basic 4.0 Commands	
APPEND	3-4
BACKUP	3-4
CATALOG (DIRECTORY)	3-5
COLLECT	3-5
CONCAT	3-5
COPY	3-6
DCLOSE	3-6
DLOAD	3-7
DOPEN	3-7
DSAVE	3-8
HEADER	3-8
RECORD#	3-9
RENAME	3-9
SCRATCH	3-10
AUTO RUN (shift RUN/STOP)	3-10
Machine Language Monitor	4-1
Entering the Monitor	4-1
The Register Display	4-1
Memory Display and Modify	4-2
Disassembling 6502 Code	4-2
Simple Assembler	4-3
Saving Memory	4-3
Loading Memory	4-3
Executing Code	4-4
Hunt memory	4-4
Fill memory	4-4
Printing disassembly	4-5
Transfer memory	4-5
Leaving the Monitor	4-5
Appendix A - Theory of operation	A-1
Appendix B - Programming considerations	B-1
Appendix C - Parallel printer pinout	B-1

Unpacking

Before installing the BusCard II in your Commodore 64 check the package you received to make sure you have everything listed below.

- The BusCard II interface

- BusCard II cable with clip

- Warranty registration card

- Parallel printer cable - **(optional)** to connect any parallel printer to the BusCard II. This is available as an option with the BusCard II or as a separate item at the dealer where you purchased the BusCard II. (Part # BC-par)

In addition, you should have on hand the following items:

- small phillips (cross) head screwdriver for opening your C-64

- PET to IEEE-488 cable to connect any IEEE peripherals (disk drive, printer, etc) to the BusCard II. Available from any Commodore dealer.

Installation

Make sure that the power for your Commodore 64 computer is **disconnected** and that any cartridges and connectors are removed. Open your computer by removing the three phillips head screws under the bottom front of the computer. Tilt the front of the cover up and back to remove the keyboard section of the computer. It will still be attached to the lower section by the keyboard and power light wires.

Just left of the centre line of the computer board and about half way back is a cylindrical device (resistor) labelled **R44.** Locate this and clip the connector on the supplied cable to the right (or bottom) end of the resistor. For best connection hook the clip under the resistor lead from front to back and lay the clip down toward the front of the computer. (Diagrams 1 and 2)

Route the cable to the back right area and pass the connector out under the metal shield surrounding the cartridge port connector.

Re-assemble the computer. Make sure that the case is fully closed and that the clips are laying down towards the front.

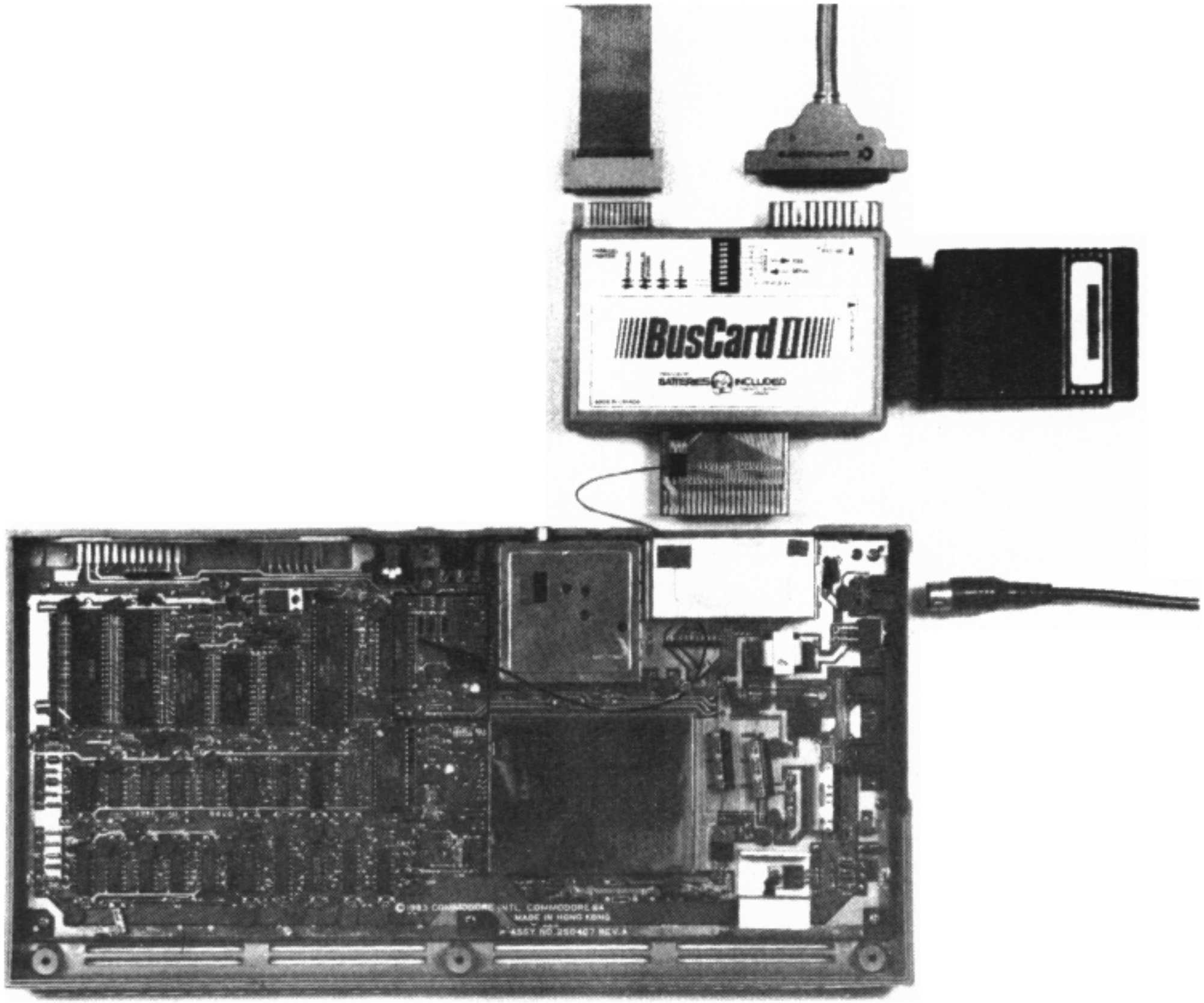
Reconnect only the power connector and display (TV or monitor) connections to the C-64 and turn it on. It should power up and display the normal sign-on message. **Turn the unit off.**

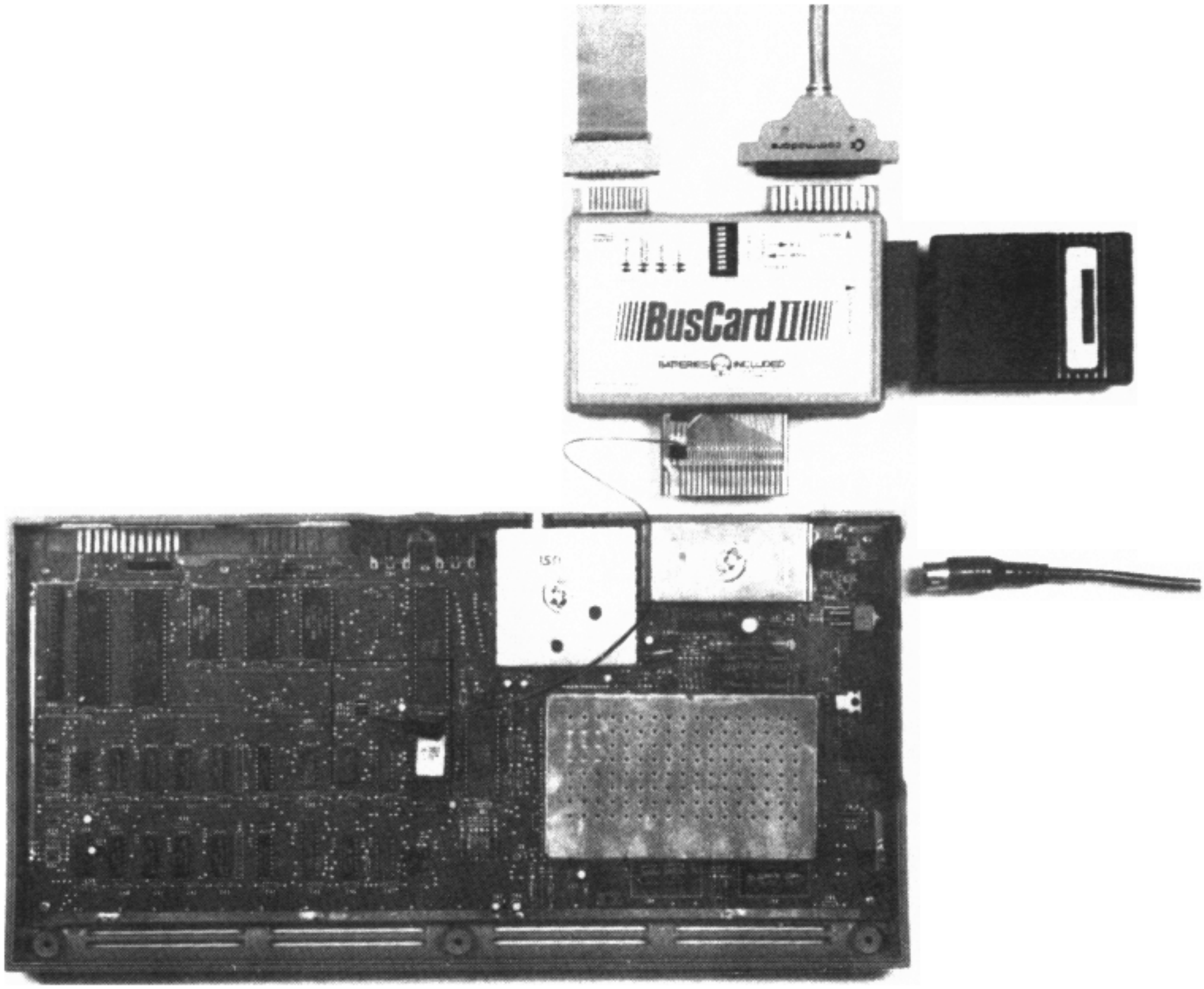
Place the BusCard II interface behind the computer and push the cable connector hanging out of the cartridge slot onto the corresponding right angle connector at the front of the BusCard II. Check that all the pins are engaged. If, after installation, you have to remove the BusCard II temporarily from your computer, the cable may be unplugged and left installed; it will not interfere with normal operation.

Insert the BusCard II into the cartridge slot of your computer. Make sure it is fully seated in the connector. Turn the computer on again. It should power up and display the normal sign-on message. Turn the unit off.

Connect any IEEE-488 peripherals to the connector at the right rear of the BusCard II. PET to IEEE cables are usually keyed and are inserted label side up.

Connect the parallel printer cable (if any) to the printer port at the left rear of the BusCard II. This cable is optional. The other end of the cable should plug directly into the parallel input connector on your printer.


BusCard II
BATTERIES INCLUDED
INTL. COMMODORE 64
MADE IN HONG KONG


BusCard II
BATTERIES INCLUDED

Switches

The switches which are located on the top of the BusCard II are used to tell the C-64 where to look to find a particular device. With these switches it is possible to have a mixed system consisting of some devices on the **IEEE-488** bus, some on the **Commodore Serial** bus, and one on the BusCard II **parallel** printer port. These switches are generally only set once for a particular system.

Device Number 4 (Printer)

Switches 1 and 2 on the top of the BusCard II control the allocation of device 4. These switches control which port will be used if a program accesses device 4. This is usually the printer and the BusCard II allows the following options in connecting your printer.

Type of device	sw1	sw2	Where to connect
IEEE-488 printer (2022/3,4022/3,8023 etc.)	OFF	OFF	BusCard II (at right)
Commodore Serial printer (1515,1525/6,MPS801)	**ON**	**ON**	C-64 Serial connector
Parallel printer (Epson,Centronics etc.)	**ON**	OFF	BusCard II * (at back)
	OFF	**ON**	

If switch 1 is **ON** and switch 2 is **OFF** device 4 is configured for a parallel printer and automatic character conversion from PET ASCII (PETSCII) to True ASCII (as used by most printers) is **enabled.** If switch 1 is **OFF** and switch 2 is **ON** the conversions are **disabled.** Conversions are performed to make upper and lower cases on the printer and C-64 screen agree. This is useful if a BASIC program which **assumes** that you have a **CBM** printer is required to print on an **ASCII** printer. **Most word processors, etc will require that conversions be turned off : they perform their own character conversions.**

Device Numbers 5-10 (Including Disk Drives)

Switches 3 to 8 on the BusCard II control the device allocation for devices 5 to 10 respectively. If a switch is **OFF** then that device is allocated to the **IEEE-488** bus on the BusCard II. If a switch is **ON** then the device is assumed to be on the **Commodore Serial** bus, available on the back of the C-64.

Switch #	Device #	
3	5	
4	6	
5	7	
6	8	(disk drive is usually 8)
7	9	
8	10	

Device Numbers 11 and up

Devices **11 and up** have no switches and are **permanently** allocated to the **IEEE-488** bus on the BusCard II.

General Configuration

You need only set the switches for the devices you attach. Other devices numbers will be ignored and will react with the normal "DEVICE NOT PRESENT" error-message if they are accessed.

You may already have an interface from the Commodore Serial or IEEE-488 buses to your parallel printer. These printer interfaces may remain connected to the bus they normally use **or** the printer may be connected directly to the BusCard II parallel port. If you have an external printer interface, set the switches to allocate the printer device number (usually #4) to the appropriate port, not necessarily the parallel port.

If you have an IEEE disk drive and a 1541 Serial disk drive and wish to connect both to your C-64, there are two options:

You may leave both drives as device 8, one on the Serial bus and one on the IEEE-488. This requires you to switch between the two buses by changing switch 6.

Alternately, you may change the device number of one of the disk drives to 9. This allows you to set up device 8 on one bus and device 9 on the other. Both Serial and IEEE disk drives may have their device numbers changed by cutting a trace in the drive. See your dealer for information on this. The IEEE disk drive models 4040, 8050, 8250, 9060 and 9090 may be temporarily changed to device 9 with the following short program:

```
10 open1,8,15
20 print#1,"m-w"chr$(12)chr$(0)chr$(2)chr$(41)chr$(73)

...use the following for the 1541 and 2031 LP.

10 open1,8,15
20 print#1,"m-w"chr$(119)chr$(0)chr$(2)chr$(41)chr$(73)

... these programs are cancelled by resetting the drive.
```

Re-reading The Switches

If you change the switch settings the changes take effect immediately. **Do not change the switch settings while the computer is actively performing <u>any</u> input or output operations.**

Basic 4.0

Basic 4.0 is a set of additional commands added to the regular set available in the standard Basic 2.0 language. This section is intended to outline and explain in simple terms the additional commands available with the Basic 4.0 language. It is not intended to teach programming or explain in detail the techniques which can utilize these commands. For more comprehensive information the following sources may be useful:

User's Reference Manual - Commodore BASIC Version 4.0
Part# 321604 - Commodore Business Machines Inc.

User's Manual for CBM 5 1/4-inch Dual Floppy Disk Drives
Part# 320899 - Commodore Business Machines Inc.

PET PERSONAL COMPUTER GUIDE
- Adam Osbourne, Jim Strasma, Ellen Strasma
Copyright 1982 - Osbourne/McGRaw-Hill - ISBN 0-931988-76-4

The following convention has been used for the Basic 4.0 command descriptions. Commands are in UPPER case, variable data is in lower case. Square brackets [and] indicate an optional parameter. Quotes "" indicate that a string constant or variable is required. Angle brackets <> indicate that a constant or variable may be used. In most cases variables must be enclosed in brackets () to avoid a SYNTAX ERROR.

Turning Basic 4.0 ON and OFF

When the computer is first powered on the Basic 4.0 language extension is not enabled. Enable it with **SYS 61000** . To disable the Basic 4.0 language and return operation to the standard Basic 2.0 language enter the command **SYS 61003** . Do NOT enable or disable the Basic 4.0 language from within a Basic program - only from the keyboard in immediate mode.

DS and DS$ (Disk Status) and ST (I/O Status)

Because the disk drive controller is a separate computer, any errors it encounters during operation must be manually retrieved from the drive to the computer. Error messages are usually indicated by a RED centre light on dual slot drives, or a rapidly flashing RED light on single slot drives. Basic 4.0 can retrieve and display an error message from the disk drive simply by referring to the pseudo-variables DS and DS$. DS and DS$ are used as if they were regular Basic variables, except that you cannot assign values to them. Instead, when you refer to either DS or DS$ after issuing a disk related command (such as DSAVE), the computer will fetch the current error status of the disk unit last referred to and store the resulting string in DS$. The first number in the DS$ string is stored in DS - this is known as the error code. It is important to note that DS and DS$ are only updated when you refer to them AND a disk related command has been issued since the last reference to DS or DS$. If you refer to DS or DS$ and a disk command has not been issued since the last reference, then the current values are reused and the drive is not queried. This allows programming of the form:
IF DS <> 0 THEN PRINT DS$: STOP

There are three messages which are not errors (and do not light the error lamp) which may be retrieved by DS and DS$.

73, cbm dos v.....,00,00 - This message waits in the drive after power-up until a command is executed.

00, ok,00,00 - This is the message returned if the previous command was executed with no errors. It is also returned when there was an error, DS or DS$ was referenced, and for some reason the drive was queried a second time without any disk command being issued (this normally shouldn't happen since DS and DS$ keep track of disk commands).

01, files scratched,xx,00 - This message is only returned after a SCRATCH command is executed by the drive. The two digits 'xx' indicate the number of files actually scratched by the drive - it will be zero if no files were scratched.

Note 1: If no device responds to Basic 4.0's request for the disk message then DS will have a value of 0 to 20. DS$ will have a length of zero.

Note 2: The act of fetching DS or DS$ will cause ST to be changed - therefore you must save ST before checking DS and DS$ if ST is needed later in the program.

STATUS - (ST) - result of last I/O operation

ST is used as if it was a regular Basic variable, except that you cannot assign a value to it. It is used as a variable which contains the result of the last attempted I/O operation. The value will vary depending on the success of the I/O with the various bits indicating exact nature of the problem.

ST numeric value	IEEE meaning
1	time out on write
2	time out on read
4	none
8	none
16	none
32	none
64	EOI (end or identify)
-128	device not present

Status values of 1 and 2 indicate that the device you are trying to communicate with is not accepting or sending data. For instance, if when reading from a sequential file the program attempts to read more data than exists in the file, the drive will TIME OUT ON READ and status will equal 2. A status value of 64 on reading from a file indictes that the last data item fetched is the last data item in the file and there are no more to follow. The computer often generates a status of 64 when it is sending data to the disk drive or printer. If status is -128, then the device number you are trying to communicate with does not exist.

DOS file name pattern matching

The Commodore DOS systems allow considerable flexibility in specifying which files a command will act upon. For example, if you wished to scratch (delete) a file or group of files, you could specify the file name in many different ways:

```
fred      -the single file 'fred'
fred*     -all files starting with the characters 'fred'
fr?d      -all files matching 'fr?d', where '?' is any character
fr?d*     -all files matching 'fr?d' with anything following
0:fred    -the file 'fred' on drive 0
1:fred*   -all files on drive 1 starting with 'fred'
fred=prg  -the program file 'fred'
0:f*=seq  -all sequential files on drive 0 which start with 'f'
1:*=rel   -all relative files on drive 1
```

Many disk commands can use some or all of the methods shown above to specify files. Others will require a specific format as described in the individual commands.

Basic 4.0 Commands

APPEND

Format - APPEND# <file number> , "<name>" [,D<x>] [ON U<y>]

Use - To add additional data to the end of a sequential disk file.

Notes - APPEND is used like a DOPEN command but can only be used to add data to an existing sequential disk file. APPEND opens the specified data file for write and positions the DOS pointers to the current end of the file and new data can be added. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8, drive 0.

Example - x=1 : APPEND#1, "data file", D(x) ON U 9
Opens a file with a logical number of 1 called "data file" on drive 1 of unit 9 for append.

BACKUP

Format - BACKUP D<x> to D<y> [ON U<z>]

Use - To create an exact duplicate of a diskette using a dual drive

Notes - Not applicable for single slot drives.
Both drive numbers must be specified and must be 0 and 1. Unit defaults to device 8. Any variable or evaluated expressions must be enclosed in parentheses.

Example - BACKUP D0 to D1

CATALOG or DIRECTORY

Format - CATALOG ["<pattern>",] [D<x>] [ON U<y>]
DIRECTORY ["<pattern>",] [D<x>] [ON U<y>]

Use - Displays on the current output device (usually the screen) the disk directory from the specified drive. If no drive is specified both drives are searched in a dual drive. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8. Pattern matching as described on Page 3-3 may be used to view limited segments of the directory. Pressing the SPACE bar will cause the scroll of the display to pause, pressing it again will continue the display. STOP terminates the listing.

Example - CATALOG D0 ON U 9
DIRECTORY D0, "fred*"

COLLECT

Format - COLLECT [D<x>] [ON U <y>]

Use - Causes the disk drive to review the block allocations for all files in the directory, deleting all incorrectly closed files (those marked with an asterisk in the directory). The block count and allocation map are updated to correspond correctly to the files currently on the disk. Space allocated by the BLOCK-ALLOCATE command is freed since it is not linked to the directory. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8, drive 0.

Example - COLLECT D0

CONCAT

Format - CONCAT [D<x>,] <"namea"> TO [D<y>,] "nameb" [ON U<z>]

Use - Causes the sequential file "nameb" to have the data in "namea" appended to it. The data in the file "namea" is unaltered. Can only be used with sequential data files. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8, drive 0.

Example - CONCAT "two" TO "one" ,D1
creates file "one" which contains "one" + "two"

COPY

Format - COPY [D<x>,] "<namea>" TO [D<y>,] "<nameb>" [ON U<z>]
or - COPY D<x> TO D<y> [ON U<z>]

Use - causes the disk drive to copy a file or files from one place to another within a disk unit. It can be used to copy data from one disk to another in a dual drive and to copy a file to another place on the same diskette. Pattern matching as described in Section 3.2, Page 7 may be used to copy multiple files. If the file is to be copied to the same diskette then "nameb" must differ from "namea". COPY without file names copies all files from one diskette to the other in a dual drive without erasing those already there. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8.

Example - COPY D0 to D1
COPY D1, "temp" TO "tempb"

DCLOSE

Format - DCLOSE [#<x>] [ON U<y>]

Use - Close one or many currently open disk files. DCLOSE will only close those disk files which were currently open in the computer's internal file table. If a logical file number <x> is specified then only that file will be closed. If only the unit number <y> is specified then all files on that unit will be closed. If both logical and unit numbers are specified then the unit number is ignored. If no parameters are given then all disk files will be closed. Relative record files may have more records generated at close time than were actually written in order to fill out the sectors in use. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8.

Example - DCLOSE :closes all files currently open on device 8
DCLOSE#2 :close the file with logical number 2

DLOAD

Format - DLOAD "<name>" [,D<x>] [ON U<y>]

Use - Load a Basic program from disk, relocating it if necessary. Any variable or evaluated expressions must be enclosed in parentheses. Pattern matching as described on Page 3-3 may be used. If an asterisk (*) is used as the filename the last accessed program on the disk will be loaded (if you exchange diskettes after the first load then this will not work). If the last file accessed was not a program file then the first program file in the disk directory will be loaded. Unit defaults to device 8. If no drive is specified both drives are searched in a dual drive. If used in program mode the subsequently loaded program will begin execution with the first line.

Example - DLOAD "first file" ,D1

DOPEN

Format - DOPEN# <a> , "<name>" [,L<y>] [,D<x>] [ON U<z>] [,W]

Use - Open a sequential or random access file for read or write. <a> is the logical file number. Logical file numbers greater than 127 cause each PRINT# statement to transmit a LINE FEED character and a CARRIAGE RETURN character. The L<y> parameter only needs to be specified when opening a relative record file for the first time. Relative record files are always opened for both read and write. If not specified, sequential files are opened for read. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8, drive 0. Both drives in a dual drive will be searched if all the optional parameters are missing. If the first character of the filename is the @ (at) symbol, it is removed and the specified file is opened replacing any existing file with that name provided that the file type is the same.

Example - DOPEN# 5 , "fred" , L 27 , D1
open a relative file, record length of 27 bytes, on drive 1.

DOPEN# 2 , "@abc file", W
opens sequential file on drive 0 for write with a logical file # of 2.
If "abc file" previously existed it has now been replaced.

DSAVE

Format - DSAVE "<name>" [,D<x>] [ON U<y>]

Use - Save a Basic program to disk. The "name" parameter can be up to 16 characters long. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8, drive 0. If the first character of the filename is the @ (at) symbol, it is removed and the specified file replaces any existing file with that name provided that the file type is the same.

Example - DSAVE "my program" ,D1
DSAVE "@freds prog"
save "freds prog" on drive 0 replacing any previous program file named "freds prog".

HEADER

Format - HEADER "<disk name>" [,D<x>] [I<zz>] [ON U<y>]

Use - To format (new) a new disk or erase the contents of an old one. The <disk name> may be up to 16 characters long. A 'STRING TOO LONG error will occur if the <disk name> is greater than 16 characters long. If the I<zz> parameter is missing then the drive will attempt to reformat only the directory track. The <zz> parameter may be specified as a string variable. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8, drive 0.

When used in immediate mode the question 'ARE YOU SURE?' will be asked before the command proceeds. Answer Y or YES to continue, N to abort. On completion the screen will display **READY** if the disk was correctly formatted or **BAD DISK** if any error was encountered. The actual error can be displayed by printing the contents of DS$.

CAUTION: The HEADER command will erase all information previously stored on the disk! Be careful!

Example - HEADER "my first disk" , D1 ON U 8 , Ibz
HEADER "next disk"

RECORD

Format - RECORD# <logical file#> , <record#> [,<byte position>]

Use - Used before a GET#, INPUT# or PRINT# statement to position to the correct record number in a random access data file opened with the <logical file number>. <record#> must be between 1 and 65535 inclusive. <byte position> must be between 1 and 254 inclusive. <byte position> defaults to 1 if omitted. Any variable or evaluated expressions must be enclosed in parentheses.

If the <record#> causes the pointer to be positioned beyond the current end of file (as it would be when adding data to the file) the disk status (DS$) will show a RECORD NOT PRESENT error. If the intent is to add data to the file, then the next PRINT# statement will cause that record and any necessary intervening records to be generated. If INPUT# is executed then a null value is returned with STATUS (ST) set to EOI (64).

Example - RECORD# 1, 402 , 5
positions to byte 5 of record 402

RENAME

Format - RENAME [D<x>,] "<old name>" TO "<new name>" [ON U<y>]

Use - Changes the name of a specified disk file. The file must not be currently open. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8, drive 0. It is not possible to RENAME a file to a name already existing on the disk.

Example - RENAME "freds file" TO "my file" , D1

SCRATCH

Format - SCRATCH "<name>" [,D<x>] [ON U<y>]

Use - Erase a disk file or files. Any variable or evaluated expressions must be enclosed in parentheses. Unit defaults to device 8, drive 0. When used in immediate mode the question 'ARE YOU SURE?' will be asked before the command proceeds. Answer Y or YES to continue, N to abort. Multiple files may be scratched at one time by using the pattern matching as described on Page 3-3.

Example - SCRATCH "my file"
erase only "my file" from drive 0

SCRATCH "fred*" ,d1
erase all files on drive 1 beginning with "fred"

AUTO RUN (shift RUN/STOP)

Format - Press the RUN/STOP key while holding the SHIFT key

Use - Pressing the RUN/STOP key while holding the SHIFT key down will cause the computer to load and run the first program on drive 0 of device 8.

The Machine Language Monitor

It is beyond the scope of this manual to teach machine language programming so the next section is necessarily superficial and only describes how to use the included monitor, and not why.

The monitor prompts with a single dot when expecting a command.

Entering the Monitor

The monitor is disabled on power-up and the following command must be issued to enable and enter it for the first time: **SYS 61006.** Once enabled, the monitor will be entered whenever the 6510 executes a BRK instruction, such as when typing SYS 8. If a RUN/STOP-RESTORE is performed, the monitor will be disabled and must be SYS'ed again to restore it. It is a good idea to enter the monitor with SYS 61006 to avoid confusion.

Register Display (R)

Upon entering the monitor, and upon execution of the 'R' command, the MLM (machine language monitor) will display the contents of some pertinent CPU registers. The register names are given on one line with their corresponding contents shown below:

```
   B*  PC  AC SR XR YR SP
   .; 04F9 2A 30 FF 54 ED
```

PC - The 6510 program counter just after the BRK instruction.
AC - The 6510 .A register (accumulator) contents.
SR - The 6510 status register (.P) register contents.
XR - The 6510 .X index register contents.
YR - The 6510 .Y index register contents.
SP - The bottom of the 6510 stack (in this case $01ED).

All values are represented in hexidecimal. The contents of any of the registers may be changed by moving back over the line with the values on it, typing over the old contents and striking 'RETURN'.

Memory Display and Modify (M)

The contents of any number of memory locations may be displayed and modified with the 'M' command. Typing 'M', followed by a space, followed by the starting address, followed by a space, followed by the ending address, will display as many lines of eight bytes as required to show the entire range. The starting and ending addresses are four hex digits. If the end address is missing, only a single line will be displayed. The ending address must be higher than the starting address.

```
.M 03F0 0402
.: 03F0 25 74 EE D4 6F AA 00 02
.: 03F8 AD C0 14 15 85 51 EE 35
.: 0400 00 12 04 0A 00 97 45 46
```

As with the register display, any of the displayed memory locations may be changed merely by typing over the two hex digits and striking 'RETURN'. All of the bytes displayed on that line are re-written to memory. In some instances, particularly when viewing the I/O chips, it may not be desirable to write to all of the bytes on a displayed line. In this case, simply write spaces over the displayed bytes which are not to be re-written when 'RETURN' is pressed: only the bytes remaining shown on the line will be written. If a '?' is displayed by the monitor after a memory byte when that line has just been re-written, it means that the previous byte was written, but when re-read to verify, was found to be in error; this may be acceptable depending on what is present at that address.

Disassembling 6510 Instructions (D)

A screen of 6510 mnemonics may be displayed for a block of memory by typing 'D' followed by the starting address. The memory address, actual memory byte contents, and the mnemonic equivalent of the instruction will be displayed for 23 sequential processor instructions. Undecodable opcodes will be shown as '???'.

Instructions shown on the screen may only be modified by re-typing the byte content field of each line, and not the mnemonic field. Sequential pages of display are possible by striking 'D' 'RETURN' on the bottom line of suceeding pages.

Simple Assembler (A)

Included in the monitor is a non-symbolic assembler which will translate 6510 mnemonics and operand fields to equivalent 6510 code bytes in memory. The format is:

.A XXXX III OOOO

...where XXXX is the instruction's address, III is the mnemonic, and OOOO is the (optional) operand field.

All values must be entered in hex and must be one or two bytes long and preceded by a '$'. Immediate operands would take the form: #$xx. Branch instructions have the destination address for an operand; the assembler will calculate the offset by itself. To obtain examples of the format of the assembler, try disassembling portions of the ROM code in the computer and examining the printed format; it is the same as that required by the assembler.

The assembler will automatically generate the address of the next instruction, so writing short segments of code is quite simple. For longer programs a symbolic assembler is recommended.

Saving Memory (S)

It is possible to save a block of memory to a storage device (disk or cassette) with the following command:

.S "name",aa,bbbb,cccc

...where 'name' is the file name including any drive number etc. 'aa' is two hex digits for the device number. 'bbbb' is the four hex digits of the starting address. 'cccc' is the four hex digits of the ending address +1.

Loading Memory (L)

It is possible to load a block of memory from a storage device with the following command:

.L "name",aa,xxxx (,xxxx is optional)

Where 'name' and 'aa' have the same meaning as above. Note that the load is normally non-relocating. If the **,xxxx** segment is present, loading will begin at the address specified.

Executing Code (G)

From within the monitor, you may directly pass control to a routine at some address with the Go command. If no address is given after the 'G' command, then the last address shown in the Register display of the program counter (PC) is the location where execution will resume. Alternately, you may specify a four hex digit address at which to start.

Hunt memory (H)

A specified section of memory may be searched looking for a specified sequence of bytes. Type 'H' followed by a space, followed by the starting address, followed by a space, followed by the ending address, followed by a space, followed by the first byte in the search string, followed by a space, followed by the second byte, etc. The starting and ending addresses are four hex digits and both must be present. The ending address must be higher than the starting address. The hunt function is also capable of searching for a string of PET-ASCII characters. To search for a string, type a single quote character followed by the search string in place of the byte sequence above.

```
          .H 040A 41BF 4C 00 A0
          .04F3 05D2 215A 41A0

or...     .H 0803 105B 'BASIC
          .A002 010E
```

The address of each area of memory that contains the specified search string is displayed, followed by a space, followed by the next address, until all occurances of the search string have been noted.

Fill memory (F)

A specified section of memory may be filled with a specific byte. Type 'F' followed by a space, followed by the starting address, followed by a space, followed by the ending address, followed by a space, followed by the byte that memory is to be filled with. The starting and ending addresses are four hex digits and both must be present. The ending address must be higher than the starting address.

```
          .F 050C 21CA FF
          .
```

Printing disassembly (P)

The P command provides a continuous disassembly of memory from a starting address to an ending address. Output is of the same form as the D command, but does not stop until the ending address is reached. Before using this command it is neccessary to open the output channel from Basic. From Basic type:
OPEN 4,4 : CMD4 : SYS 61006

Once in the monitor, type 'P' followed by a space, followed by the starting address, followed by a space, followed by the ending address. The disassembly output will be directed to the printer. When the printing has stopped, exit the monitor by typing 'X' and 'RETURN'. Then type PRINT#4 and 'RETURN'.

```
.P C000 C504
.X
READY
PRINT#4
```

Printed output from the other functions of the monitor can be accomplished in the same way by typing the appropriate command in place of the 'P' instruction.

Transfer memory (T)

A specified section of memory may be transferred to another area in memory with the 'T' command. Type 'T' followed by a space, followed by the starting address, followed by a space, followed by the ending address, followed by a space, followed by the new starting address. The starting and ending addresses are four hex digits and both must be present. The ending address must be higher than the starting address. The original area of memory is unchanged. Care must be taken when the new memory area overlaps the old area.

```
.T A000 BFFF 4000
.
```

Leaving the Monitor (X)

Typing 'X' followed by 'RETURN' will exit the monitor and return to Basic.

The BusCard II operates by permanently mapping out a section of the internal KERNAL ROM in the C-64. All of the normal Serial bus routines are present in the segment of the KERNAL ROM from $EC00 to $EFFF. The BusCard II ROM replaces this segment of KERNAL in order to dispatch the low-level commands to the proper device handler depending on the switch allocation of that device number. The actual device handlers (IEEE, Serial bus, Parallel printer), machine language monitor and Basic 4.0 are switched into the memory space only when required. The image appears in place of the RAM from $C000 to $CFFF. NMI and IRQ traps have been implemented so that the BusCard II ROM will not interfere with access to code at $C000. The machine language monitor may not be used to examine memory in the KERNAL ROM from $EC00 to $EFFF.

If the KERNAL ROM image is transfered back into the RAM behind $E000-FFFF and the ROMs disabled, the ROMs are temporarily restored for any I/O operations by the BusCard II.

The input/output chips used in the BusCard II are 6532A and 6520A mapped into the I/O space from $DE00 to $DEFF. The I/O space is normally open and the BusCard II I/O chips are switched in only for the actual I/O operation. The entire I/O space is available to any other device which might be plugged into the BusCard II cartridge extension.

The connection made using the clip inside the C-64 is used to obtain the HIMEM control bit from the 6510 I/O port. This allows the complete invisibility of the BusCard II by ensuring that the interface ROMs disappear and appear using the same controls as the internal ROMs. If the wire is not connected, all BusCard II functions will still operate with the exceptions that the RAM behind $EC00 to $EFFF will not be available and some cartridge software may not operate.

Appendix B Programming considerations

Because of the efforts taken to ensure maximum invisibility, very few precautions need be taken when programming with the BusCard II installed.

All machine language input/output operations should be done through the KERNAL ROM routines. They may be called directly but the preferred method uses the Commodore jump table. None of the Serial routines should be entered in the middle; the BusCard II only traps the normal entry points.

Appendix C Parallel printer connection

```
BusCard II       Signal              Printer
par. port

 1..............Strobe..............1
 3..............Data 0..............2
 5..............Data 1..............3
 7..............Data 2..............4
 9..............Data 3..............5
11..............Data 4..............6
13..............Data 5..............7
15..............Data 6..............8
17..............Data 7..............9
19..............N/C.................10
21..............Busy................11
23..............N/C.................12
25..............N/C.................13

 2..............GND.................19
 \               \                   \
22..............GND.................30
```