Utilink

CARTRIDGE

FOR THE COMMODORE 64

A software cartridge that allows the user to take advantage of the power inside the Commodore 64 computer and the 1541 Disk Drive by instantly adding to the computer...

10 DOS Support Commands

16 Program Support Commands

24 Interrupt Support Commands

3 Data Conversion Commands

PRELIMINARY DOCUMENTATION

Missing Link Products

COMPUTER PERIPHERALS & SOFTWARE

PO Box 6460
Colorado Springs, CO
80934

(303) 473-8909

UTILINK CARTRIDGE

by Dennis Kerrisk

May 1983

UTILINK USER'S GUIDE

by Robert Steinbeiser

June 1983

Missing Link Products
PO Box 6460
Colorado Springs Colorado 80934
United States of America
(303) 473-8909

TABLE OF CONTENTS

INTRODUCTION

If you have spent any time at all using the Commodore 64 computer with the 1541 disk drive, you may or may not have realized all the power your system possesses. The 64 is a remarkable machine with all kinds of "bells and whistles" that make it very interesting to program. The 1541 disk drive actually contains a computer within itself to control the way it stores and manipulates your programs and data. Together these two devices form a system that can offer the user a myriad of software and hardware applications.

It seems the more powerful a system is, the more complex it becomes to operate. The user soon finds that to take advantage of the finer points of the 64, he is soon lost in memory maps, peeks, and many pokes. This new "intelligent" disk drive can do all sorts of tricky things, if you have all those device #s, file #s, opens, and print #s correct or all that hardware intelligence is soon lost on the confused user.

This is why the Utilink was created. The Utilink is a cartridge (ie: read only memory) containing a software program that installs itself firmly inside the 64 immediately after you plug it in and turn on the computer. This program then, resident in the "conciousness" of your machine, turns many of those often used pokes and peeks into simple keystrokes. It allows you to communicate with the 1541 disk drive with a few characters instead of the small programs it took before.

The Utilink actually adds 10 single character disk commands, 16 program support commands, 3 data conversion functions, and 24 interrupt supported (available anytime) commands to your computer. With a little practice, you will find the Utilink an invaluable tool that greatly enhances the effectiveness of your computer.

LIST OF UTILINK FUNCTIONS

1.0 DOS Support Commands

The disk operating system provided on the Utilink cartridge is based on the "DOS Manager" by Bob Fairbairn, Commodore Business Machines Inc.

DOS support commands allow the user to send instructions to the disk drive with short one or two character commands. All of the functions of the Utilink are active as soon as the 64 is turned on with the Utilink installed.

1.1 TO LOAD A PROGRAM:

To load a program from the 1541 drive into BASIC memory, use the " / " (backslash) character followed immediately by the name of the program to be loaded from disk.

example: /TEST <return> would load the program named "TEST" from the disk drive into the top of available BASIC memory.

If a disk error should occur, the error number, error description, track, and sector will be displayed on the screen.

1.2 TO LOAD A PROGRAM AND NOT RELOCATE IT:

To load a program from the disk drive to the exact memory locations it was saved from, use the " % " character followed immediately by the name of the program to be loaded from disk. When a program is loaded with the backslash command it is loaded normally, that is at the top of available BASIC memory. When a program is loaded with the " % " command it is loaded into exactly the same memory location it came from. This function is used primarily with machine language programs.

example: %TEST <return> would load the program named "TEST" from the disk into the exact place in memory from which it came.

This is identical to LOAD "test",8,1 in an unenhanced 64.

1.3 TO LOAD AND RUN A PROGRAM:

To load a BASIC program from disk and have it immediately run, use the " ↑ " (up arrow) character.

example: ↑ TEST <return> would load the BASIC program named "TEST" from disk and run it immediately.

1.4 TO SAVE A PROGRAM:

To save a program from BASIC memory to the disk drive, use the " ← " (left arrow) character followed immediately by the name of the program.

example: ← TEST <return> would store the program from BASIC memory to the disk with the name "TEST".

To replace a program on disk with another program of the same name, the " @: " characters are placed after the " ← " character and before the program name.

example: ← @:TEST <return>would store the program from BASIC memory to the disk and replace any other program named "TEST".

If a disk error should occur, the disk error number, disk error, track, and sector will be displayed on the screen.

1.5 TO DISPLAY THE DISK DIRECTORY:

To display the directory of the disk currently in the drive, use the <shift> D characters.

example: <shift> D <return> would display the directory on the screen.

The directory is displayed only, and any BASIC program in memory is not disturbed. (To change the color of the characters in the directory, see section 3.5.)

1.6 TO DISPLAY THE DISK NAME, ID, AND FREE DISK BLOCKS:

To display heading and free space information of the disk currently in the drive, use the " < " character.

example: < <return> would display the disk name, ID, and amount of free blocks.

1.7 TO READ THE DISK ERROR CHANNEL:

To determine the cause of a disk error signified by the flashing red LED on the front of the disk drive, use the " @ " character.

example: @ <return> would display the disk error number, disk error, track, and sector on the screen.

1.8 TO DISPLAY THE UTILINK START UP MESSAGE AND VERSION:

Use the " = " character.

example: = <return> would display the start up message and version of the Utilink cartridge in use.

2.0 OTHER DISK COMMANDS:

A detailed list of all the commands that the 1541 disk drive will accept may be found on pages 14-41 of the VIC-1541 Disk Drive Users Manual. Below are some of the more common commands, but any command can be sent to the drive in a similar manner. In the below commands, the drive number (0 or 1) has been omitted. The drive number is only used with systems that contain a dual disk drive.

2.1 TO FORMAT A NEW DISKETTE:

Before a new diskette can be used in the drive, it must first be formatted. To do this use the the " @ " character followed by the " N " disk command.

example: @N: disk name,ID <return>

The disk name is the name of the entire diskette. The ID can be any two characters. It is stored on the directory and also every header throughout the disk.

2.2 TO COPY A PROGRAM OR FILE:

To duplicate any file or program on a disk, use the " @ " characters followed by the " C " disk command.

example: @C:new file=old file <return>

This command will only duplicate files or programs on the same disk. It will not copy from one disk to another.

2.3 TO REMOVE A PROGRAM OR FILE:

To remove a program or file from the disk, use the " @ " character followed by the " S " disk command.

example: @S:file name <return>

2.4 TO INITIALIZE THE DISK DRIVE:

To clear the drive of any errors and return it to the same state as when powered up, use the " @ " character followed by the " I " disk command.

example: @I <return>

2.5 TO VALIDATE A DISKETTE:

To validate a diskette, use the " @ " character followed by the " V " disk command.

example: @V <return>

If a diskette has been in use for some time, the directory may become disorganized. Small amounts of disk space may be wasted. The "Validate" command reorganizes the disk to take advantage of all available space. For more information on this command, see the 1541 Users Manual.

2.6 TO RENAME A PROGRAM OR FILE:

To change the name of a program or file on the disk, use the " @ " character followed by the " R " disk command.

example: @R:new file name=old file name

2.7 TO DISABLE ALL UTILINK FUNCTIONS:

To disable all Utilink functions (except the interrupt supported commands), use the " q " (unshifted Q) character.

example: q <return> would disable all Utilink DOS and program support functions.

Interrupt support functions will be active after a RUN-STOP/RESTORE is performed. To enable the Utilink again, enter sys 32777 <return>

PROGRAM SUPPORT COMMANDS

3.0 PROGRAM SUPPORT COMMANDS:

The program support functions of the Utilink are short commands that aid the programmer in developing software. It is advisable that the programmer be familiar with, or have access to, the "Commodore 64 Programmer's Reference Guide" to take full advantage of these commands. Chapters and page numbers will be referenced whenever relevant.

3.1 TO LIST A PROGRAM:

To list the BASIC program currently in memory, use the " £ " (pound sign).

example: £ <return> would list the program in BASIC memory (this is 3 characters easier than the usual LIST command).

3.2 TO DISPLAY MEMORY POINTERS:

To display pointers that tell exactly where the character screen, screen editor, VIC bank, and character base are currently located, use the <shift> M keys.

example: <shift> M <return> would display on the screen four variables and their values (normal values are: E=4 V=0 S=4 B=2)

The letters displayed represent: E=screen editor pointer, V=VIC bank pointer, S=screen memory pointer, and B=character base memory pointer. The VIC (V) bank pointer will be between 0 and 3, pointing to one of the four possible 16K banks of memory that the VIC (Video Interface Chip) gets all its information from. The screen editor and screen memory pointers (E and S) represent one of the 16 possible 256 byte pages of memory in the current 16K VIC bank that are being used as screen memory or where the screen editor is assigned. The character base memory pointer (B) points to one of the 8 possible 2K blocks of memory that the character base memory is at. See pages 101-104 in the Programmer's Reference Guide for more information on pointers. The following formulas can be used to find the starting addresses for the screen memory & editor, VIC bank, and character base memory.

```
Current VIC bank          = V*(16384)
Current screen memory     = V*(16384)+S*(256)
Current screen editor     = V*(16384)+E*(265)
Current character base    = V*(16384)+B*(2048)
```

3.3 TO INCREMENT AND DECREMENT THE SCREEN MEMORY AND SCREEN EDITOR:

To increment the screen memory and editor, use the <shift> E key. To decrement the screen memory and editor, use the <shift> C key.

example: <shift> E <return> would increment the screen memory and editor one page.

<shift> C <return> would decrement the screen memory and editor one page.

The screen memory and editor can be set to any one of the four 256 byte pages of memory inside the current 16K VIC bank. Care should be taken when screen memory comes from zero page, (altering a key location could cause the computer to crash in confusion) or BASIC RAM area (changing characters on the screen here could alter any program in the BASIC RAM resulting in some interesting effects). The display pointer function can be used to keep track of where the screen memory and editor are at any time (section 3.2).

3.4 TO RESTORE A BASIC PROGAM AFTER NEW:

To restore a BASIC program after the "NEW" command has been executed, use the Old command (shift only the "O").

example: <shift> O <no shift> ld <return> would retore any program to BASIC memory after the NEW command has been executed.

This function will also work after a cold start (Reset, section 3.8).

3.5 TO SET DEFAULT SCREEN PARAMETERS:

To set the default screen, border, main, and alternate character colors, use the " & " key.

example: & <return> would allow you to choose colors for the screen, border, main, and alternate character colors.

Once the colors are set with this command, they will then return after a RUN-STOP/RESTORE or the Restore Default Screen Parameters command is executed. The alternate character color is used in the data conversion functions and disk directory displays.

3.6 TO RESTORE DEFAULT SCREEN PARAMETERS:

To restore the screen, border, character, and alternate character colors, use the " ! " key.

example: ! <return> would return the screen, border, character, and alternate character colors to those previously set.

3.7 TO ENABLE & DISABLE AUTOMATIC KEY REPEAT:

To enable auto key repeat, use the <shift> K keys. To disable auto

key repeat, use the <shift> J keys.

example: <shift> K <return> would enable auto key repeat.

<shift> J <return> would disable auto key repeat.

Automatic key repeat is a function that allows characters to be put on the screen as long as that particular key is held down (helpful when designing graphics displays using the keyboard graphic characters). When auto key repeat is enabled, there is a slight delay after the first character appears on the screen, to eliminate unintentional double keying.

3.8 TO CAUSE A POWER UP RESET:

To cause the C-64 to completely reset itself, as if after power up (cold start), use the Reset command (shift only the R).

example: <shift> R <no shift> eset <return> would completely reset the computer.

Note: Any BASIC program in memory will appear to be lost after this command is executed, but the "Old" command (section 3.4) will restore the program in most cases. The Reset function is identical to SYS 64738

3.9 SPLIT SCREEN MODE:

Split screen mode is used to display two different screens (screens at different places in memory) on the screen at the same time. These two screens can be any combination of text or high resolution areas, within the same 16K VIC bank. This function is especially useful when developing programs that use a hi-res screen. Usually it is impossible to see screen messages such as syntax errors, values of variables etc., when the hi-res mode is enabled. The split screen allows a portion of the text screen to be viewed along with the hi-res screen. For commands that support split screen mode, see sections 3.10-3.13.

3.10 TO SET SPLIT SCREEN PARAMETERS:

To set the top split screen parameters, use the <shift> S keys. To set the bottom split screen parameters, use the <shift> T keys.

example: Set up desired screen for the top part of the split screen, then <shift> S <return> to assign those parameters.

Set up desired screen for the bottom part of the split screen, then <shift> T <return> to assign those parameters.

The parameters for split screen mode refer to the type of screen (hi-res or normal text) and the memory location of the particular

screen to be assigned. The only limitation is that both screens must be within the same 16K VIC bank.

3.11 TO ENABLE & DISABLE SPLIT SCREEN MODE:

To enable the split screen mode, use the <shift> B keys. To disable the split screen mode, use the <shift> A keys.

example: <shift> B <return> would enable split screen mode.

<shift> A <return> would disable split screen mode.

Note: Although split screen mode may be enabled, the screen may not be visibly split, see the next section on adjusting the split screen.

3.12 TO ADJUST THE SPLIT SCREEN:

Once the split screen mode is enabled, the way the screen is split can be adjusted using the F5, F7, and either the CTRL or the Commodore logo keys. The split screen can be adjusted so that a small amount of a text screen could be placed anywhere on a hi-res screen or vise-versa.

example: <Commodore logo> F5 would move the top part of the split screen down.

<Commodore logo> F7 would move the bottom part of the split screen down.

<CTRL> F5 would move the top part of the split screen up.

<CTRL> F7 would move the bottom part of the split screen up.

3.13 TO SET THE SCREEN EDITOR IN SPLIT SCREEN MODE:

When using two text screens in split screen mode, the screen editor can be set to either the top or bottom screen.

example: <CTRL> Y would set the screen editor to the top screen.

<CTRL> D would set the screen editor to the bottom screen.

INTERRUPT SUPPORTED FUNCTIONS

4.0 INTERRUPT SUPPORTED FUNCTIONS:

Interrupt supported functions are available to the user any time, even if some other BASIC program is running at the same time. These functions are implemented with one or two keystrokes, and they don't need to be terminated with <return>.

4.1 TO CHANGE THE SCREEN, BORDER, CHARACTER, AND ALTERNATE CHARACTER COLORS:

To change the color of the screen, border, or characters, use the function keys F1, F3, F5, and F7.

4.2 Press the F1 key to increment the alternate character color (this color is used in split screen, multi-color, and extended color modes).

4.3 Press the F3 key to increment the regular character color.

4.4 Press the F5 key to increment the screen color.

4.5 Press the F7 key to increment the border color.

Each time the desired key is depressed, the color is incremented to the next consecutive color.

4.6 TO SOUND THE BELL:

To output a short audio tone from the C-64, use the F8 key.

example: Pressing the F8 key would produce a bell tone.

This function is similar to the control G function on most computer terminals.

4.7 TO INCREMENT AND DECREMENT THE SCREEN POINTER:

To set the text screen to one of 16 possible locations in the current VIC bank, use the F6 key to increment the screen pointer by 256 bytes and the F7 key to decrement the screen pointer by 256 bytes.

example: Press the F6 key to increment screen memory.

Press the F4 key to decrement screen memory.

Once the text screen is changed, the screen editor can't be used unless it is also changed to the same memory location (see section 3.3). The <shift> M function can be used to keep track of which text screen is currently in use (see section 3.2). For more information on

screen memory, see pages 102-103 in the Programmer's Reference Guide.

4.8 TO ENABLE AND DISABLE THE HIGH RESOLUTION GRAPHICS SCREEN:
To enable the hi-res graphics screen, use the <CTRL> F3 keys. To disable the hi-res screen, use the <Commodore logo> F1 keys.

example: Pressing the <CTRL> and F3 keys would turn on the hi-res screen.

Pressing the <Commodore logo> and F1 keys would turn off the hi-res screen.

Once in the hi-res mode the hi-res screen, (which is the same as the character base in normal text mode) and the VIC bank can be moved (see sections 4.13 and 4.14). The disable hi-res function does only that, if the character base has been changed it will not return to normal with this function. Use the <CTRL> A or <CTRL> C function (see section 4.13 or 4.15). For more information on the hi-res mode, see pages 122-127 in the Programmer's Reference Guide.

4.9 TO CLEAR THE HIGH RESOLUTION GRAPHICS SCREEN:
To clear the hi-res screen starting at address 8192 ($2000), use the <CTRL> F1 keys.

example: Pressing the <CTRL> and F1 keys would clear the normal hi-res screen.

This function is identical to POKEing 0 into every memory location from 8192 to 16191.

4.10 TO ENABLE AND DISABLE EXTENDED COLOR MODE:
To enable the extended color mode, use the <shift> and <CTRL> and F5 keys. To disable the extended color mode, use the <Commodore logo> and F3 keys.

example: Pressing the <shift> <CTRL> F5 keys would enable extended color mode.

Pressing the <Commodore logo> F3 keys would disable extended color mode.

Extended color mode is used to control the background color of each individual character on a text screen. See page 120 in the Programmer's Reference Guide for more information on extended color mode.

11 TO ENABLE MULTI-COLOR MODE:
To enable multi-color mode, use the <shift>, <CTRL> and F7 keys.

example: pressing the <shift> <CTRL> F7 keys would turn on multi-color mode.

Multi-color mode is used to define up to four different colors within one character space in normal or hi-res mode. See pages 115-119 in the Programmer's Reference Guide for more information.

12 TO INCREMENT COLOR #1 AND COLOR #2:
To increment color #1, use the <shift>, <CTRL> and F1 keys. To increment color #2, use the <shift>, <CTRL> and F3 keys.

example: Pressing the <shift> <CTRL> F1 keys would increment color #1.

Pressing the <shift> <CTRL> F3 keys would increment color #2.

When using multi-color or extended color modes, three colors must be specified: color #0, color #1, and color #2. Color #0 is always the same as the screen color. Colors #1 and #2 can be incremented to the desired color with this function. See page 117 in the Programmer's Reference Guide for more information.

13 TO INCREMENT THE CHARACTER BASE MEMORY:
To increment the character base memory pointer, use the <CTRL> and A keys.

example: Pressing the <CTRL> A keys would increment the character base memory pointer.

The character base pointer tells the video chip where to get the data for screen characters in normal text mode and bit map data in hi-res mode. Character memory can be any one of the 8, 2K blocks in the 16K VIC bank. Since the hi-res screen is 8000 bytes long, <CTRL> A must be pressed four times to change from one hi-res screen to the other. To keep track of which character base is in use at any time, use the <shift> M function (section 3.2). See pages 103-104 in the Programmer's Reference Guide for more information.

14 TO INCREMENT AND DECREMENT THE VIC BANK:
To increment the VIC bank, use the <CTRL> and Z keys. To decrement the VIC bank, use the <CTRL> E keys.

example: Pressing the <CTRL> Z keys would increment the VIC bank.

Pressing the <CTRL> E keys would decrement the VIC bank.

The VIC (video interface chip) in the C-64, can get its data from any one of the 4, 16K blocks in RAM. This function allows the user to let the VIC use any of these sections in RAM. To keep track of which VIC bank is currently in use, use the <shift> M function (section 3.2). See pages 101-102 in the Programmer's Reference Guide for more information.

4.15 TO RESTORE THE VIDEO INTERFACE CHIP (VIC):

To restore the VIC to the same state it was after power up, use the <CTRL> and C keys.

example: Pressing the <CTRL> C keys would reset the VIC to normal parameters.

This function will instantly restore the VIC to the normal text screen, at the normal VIC bank, with the normal character base memory. It turns off hi-res, multi-color, and extended color modes but leaves the screen, border, and character colors the same.

DATA CONVERSION FUNCTIONS

5.0 DATA CONVERSION FUNCTIONS:

When developing programs, it is often necessary to convert between decimal, hexadecimal, and ASCII number systems. These functions allow this to be done instantly. Decimal numbers must be less than 65536. Hexadecimal numbers must have four digits. Leading zeros must be used to convert hex numbers with less than four digits.

5.1 TO CONVERT A DECIMAL NUMBER TO HEXADECIMAL, ASCII:

To convert a decimal number to its hexadecimal and ASCII equivalent, use the" # " character followed immediately by the decimal number to be converted.

example: #65 <return> would display..

$0041 #65 "@ A

This is the hexadecimal conversion, the original decimal number, and the ASCII conversion.

5.2 TO CONVERT A HEXADECIMAL NUMBER TO DECIMAL, ASCII:

To convert a hexadecimal number to its decimal and ASCII equivalent, use the " $ " character followed immediately by the hexadecimal number to convert.

example: $0043 <return> would display..

$0043 #67 "@ C

This is the original hex number, the decimal conversion, and the ASCII conversion.

5.3 TO CONVERT AN ASCII CHARACTER TO ITS DECIMAL AND HEXADECIMAL EQUIVALENT:

To convert an ASCII character to its decimal and hex equivilent, use the ' " ' character followed immediately by the ASCII character to convert.

example: "A <return> would display..

$0041 #65 "@ A

This is the hex conversion, the decimal conversion, and the original ASCII character.

Note: When converting hex or decimal numbers to ASCII, the conversion

is done one the hex number. The two high bytes equal on character and the two low bytes equal one character (the ASCII conversion of 0 is the @ character). This function is useful in finding the chr$ number of any character (chr$=decimal conversion).

PROGRAMMER'S NOTES

6.0 MEMORY USAGE:

The Utilink cartridge, unlike most other ROM cartridges for the C-64, takes only about 4K of RAM from the computer. While other cartridges take all of the 8K section of RAM from $8000 to $9FFF, the Utilink uses only the section of memory from $8000 to $90FF. Although the remaining memory space from $90FF to $9FFF isn't available to BASIC, it is available to the user for machine language routines, etc..

6.1 USING THE DOS SUPPORT COMMANDS FROM A BASIC PROGRAM:

To use the DOS support commands from inside a BASIC program, use the " @ " character on a program line followed by the DOS command in parenthesis.

```
example:        10 @"I"
                20 @"S:TEST"
                30 @"V"
```

This program would execute as folows...

line 10 would initialize the disk drive
line 20 would remove the program named "TEST" from the disk
line 30 would validate the disk

Any BASIC programs that use Utilink functions may be saved to disk like any other BASIC program. When later recalling these programs, the Utilink must be installed for them to run properly.

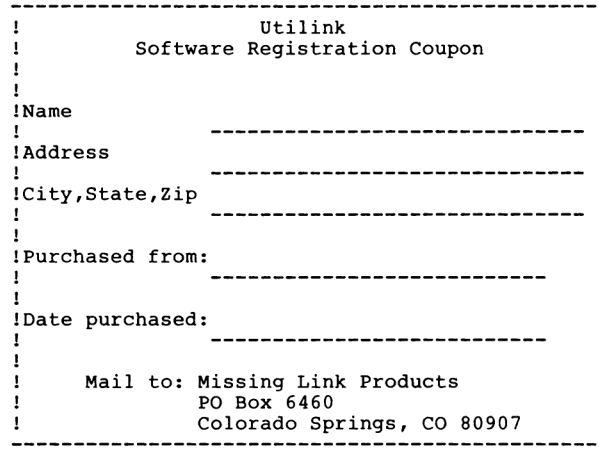
6.2 SOFTWARE SUPPORT:

Missing Link Products will provide support to the registered owner of this software. To register, complete and mail the registration coupon in the back of this manual as soon as possible after purchase.

We also welcome your comments and suggestios that may improve any new versions of this software.

INDEXED COMMAND LIST

INDEXED COMMAND LIST (cont)


Utilink
Software Registration Coupon
Name
Address
City,State,Zip
Purchased from:
Date purchased:
Mail to: Missing Link Products
PO Box 6460
Colorado Springs, CO 80907