COMMODORE 64 SOFTWARE

PROGRAMMER'S UTILITIES

UTL 6440

This package contains the following 12 utilities programs

1. change disk	7. sprite editor
2. copy-all64	8. dos wedge
3. hex dump	9. pet emulator
4. load addr	10. 1541 backup
5. supermon64.v1	11. editor64
6. char editor	12. sidmon

commodore COMPUTER

PROGRAMMER'S UTILITIES

UTL 6440

This package contains the following 12 utilities programs

1. change disk
2. copy-all64
3. hex dump
4. load addr
5. supermon64.v1
6. char editor
7. sprite editor
8. dos wedge
9. pet emulator
10. 1541 backup
11. editor64
12. sidmon

COMMODORE BUSINESS MACHINES (UK), LTD.
675 Ajax Avenue
Trading Estate
Slough, Berkshire SL1 4BG
ENGLAND

TABLE OF CONTENTS

SECTION ONE INTRODUCTION TO THE PROGRAMMER'S UTILITIES

1.1 INTRODUCTION

The COMMODORE 64 Programmer's Utilities helps the experienced and the inexperienced programmer to use our new and exciting personal computer, the Commodore 64. These programs feature the many capabilities of the Commodore 64.

On the COMMODORE 64 Programmer's Utilities diskette you will find these programs:

1. change disk
2. copy-all64
3. dump
4. load addr
5. supermon64.v1
6. char editor
7. sprite editor
8. dos wedge
9. pet emulator
10. 1541 back up
11. editor64
12. sidmon

This manual is categorised into sections demonstrating the wide variety of applications for the Commodore 64:

Section 2 – Utilities
Section 3 – Graphics
Section 4 – Sound
Section 5 – BASIC Programming Aids

1.2 USER CONVENTIONS

It is recommended that you familiarise yourself with the Commodore keyboard. Here is a brief description of certain keys and symbols, and their respective function in reference to this software.

SHIFT	To input the upper case convention of a letter, press and hold the SHIFT key in conjunction with the desired key. Then, release both keys at the same time.
RETURN	To continue the program after a line of input, press the RETURN key.
(R)	Also indicates the RETURN key.
FULL STOP	The notation for the FULL STOP (.) symbol.
SPACE BAR	The notation for pressing the SPACE BAR.
STOP	The notation for the unshifted RUN/STOP key.
GREATER THAN	The notation for the GREATER THAN (>) sign.
LESS THAN	The notation for the LESS THAN (<) sign.
UP ARROW	The notation for the UP ARROW KEY (↑).
LEFT ARROW	The notation for the LEFT ARROW key (←).
CONTROL	The notation for the CTRL key.
CURSOR RIGHT	The notation for the unshifted CURSOR LEFT/RIGHT key, i.e. the horizontal cursor control key.
CURSOR LEFT	The notation for the shifted CURSOR LEFT/RIGHT key.
CURSOR DOWN	The notation for the unshifted CURSOR UP/DOWN key, i.e. the vertical cursor control key.
CURSOR UP	The notation for the shifted CURSOR UP/DOWN key.
DEL	The notation for the unshifted INST/DEL key.
INST	The notation for the shifted INST/DEL key.
HOME	The notation for the unshifted CLR/HOME key.
CLR	The notation for the shifted CLR/HOME key.
HASH	The notation for the hash (#) sign.
ENGLISH POUND	The notation for the English Pound (£) sign.
F1 - F8	The notation for the function keys, F1 through F8. Please note that the even numbered keys are obtained by holding down the shift key and pressing the relevant function key.

SECTION TWO
UTILITIES

2.1 C64 MENU

The C64 MENU program assists you in LOADing and executing other programs that exist on the diskette. To load this file, the following procedure is recommended:

1. Turn on the Commodore 64 and 1541 disk drive

2. Place the Programmer's Utilities diskette into the drive

3. Type LOAD" *",8(R) (Note the space after the first quotation mark .)

This loads the first program on the disk, in this case "C64 MENU".

4. Wait until the screen says READY

5. Type RUN(R)

After step five has been completed the screen goes blank temporarily while the program is loading the DIRECTORY from the disk. When this task has been completed, the screen displays the directory listing and the program is now ready for a command to be entered. The following four options are available.

a. Display the next page of the directory listing
b. Create a SEQ FILE of the directory (named DIRECTORY)
c. Exit the program
d. Load and execute a program.

The first three options above require only one keystroke. For the fourth option, you must enter the number of the screen menu that is beside the desired program name, followed by a RETURN. After executing any program, you can return to the menu by following steps 3 to 5.

NOTE
All directory entries are listed in alphabetical order. Directory entries with file names which begin with a space WILL NOT be listed. These programs require special loading instructions. Please refer to the back of this manual for the exact program name.

2.2 CHANGE DISK

The Change Disk program allows you to change the "device number" of the disk drive. Its main purpose is as a companion to the COPY-ALL64 program to copy files between two drive units (i.e., two 1541s or a 4040 drive and an 8050 drive). Normally, all drive units are device 8.

To use this program:

1. Connect both drive units to your computer
2. Turn on the power of just ONE drive
3. Insert the Programmer's Utilities diskette into that drive
4. Type LOAD"CHANGE DISK",8(R)
5. Type RUN(R)
6. The program asks "OLD DEVICE NUMBER?"
7. Enter the number 8(R)
8. The program asks "NEW DEVICE NUMBER?"
9. Enter the number 9 and press RETURN - this drive is now device 9
10. Power on the other drive (device 8).

You can now load and run COPY-ALL64 to copy from unit 8 to unit 9 (or vice versa).

2.3 COPY-ALL64

The COPY-All64 program allows you to copy one or more files from one drive unit to another (see also CHANGE DISK).

To use this program:

1. Run CHANGE DISK (if necessary)

2. Type LOAD"COPY-ALL64",8 RETURN. (If you ran CHANGE disk, enter LOAD"COPY-ALL64",9 RETURN.)

3. Remove the Programmer's Utilities diskette from the disk drive and put in the diskette you want to copy (this is the "source" disk)

4. Place the disk onto which you want to copy into the other drive (this is the "destination" disk)

5. Enter RUN(R)

6. At the prompt, FROM UNIT?8 enter the device number of the source disk and press RETURN. To accept the prompted 8 simply press RETURN

7. At the prompt TO UNIT?9 enter the device number of the destination disk and press RETURN.

8. On the PATTERN prompt, enter (R) if you don't wish to use pattern matching. To pattern match enter as many characters as you want, followed by an asterisk (*), (i.e. only files whose name begin with the specified characters is selected for copy). If no asterisk (*) is used, only the file whose name exactly matches this pattern of characters is selected.
9. The program will then display each of the filenames in turn. For the files you want copied, enter Y, otherwise N.
10. The program asks if it needs to "NEW" the destination disk. If it is a new disk, enter Y. Otherwise enter N.
11. After copying is done the program will ask if there is another disk to be copied, if yes, enter Y otherwise enter N.

WARNING
NEWING THE DESTINATION DISKETTE WILL PERMANENTLY ERASE ALL INFORMATION ON THAT DISKETTE.

2.4 1541 DISK BACKUP

The 1541 Disk Backup program allows the user to backup (copy) an entire disk without the aid of a dual disk drive. The backup operation is accomplished by exchanging the input and output diskette in the disk drive.

To use this program:

1. Type LOAD"1541 BACKUP",8(R)
2. When the program is loaded, enter RUN(R)
3. An input screen is displayed and the program guides you through entry. Follow the instructions in the OPERATOR INTERVENTION block shown at the bottom of the screen.

There are two Backup commands:

The 'B' option specifies Block Allocation Map (BAM) backup. This selection determines which areas of the disk are written on and only copies those areas. (This is the quicker of the two methods.)

The 'D' option specifies duplication and produces an exact track and sector duplication of the diskette to be copied.

2.5 DUMP

The Dump program displays a disk file on the screen in hexadecimal format, ten bytes per line. Note that the line addresses shown on the left are in decimal.

1. To load the program type LOAD"DUMP",8(R)
2. To run it, enter RUN(R)
3. Program prompts for file name
4. Enter the file name and press RETURN
5. The program displays file contents.

Here is a sample display:

```
0000: 08 50 00 00 00 00 00 00 00 00
0010: 00 00 00 00 00 00 00 00 00 00
```

2.6 LOAD ADDRESS

The Load Address program is similar to Dump, but will display the load address of any program (PRG) file on the screen in decimal.

The load address is the memory address where the program started when it was saved on disk. Normally, for program (PRG) files on the COMMODORE 64, this is 2049. Some data files are also saved in PRG format (e.g., sprites) and will have various load addresses.

1. To load the program, type LOAD"LOAD ADDR",8(R)
2. To run it, enter RUN(R)
3. Program will prompt for filename
4. Enter your selection
5. The program displays load address in decimal.

2.7 SUPERMON64.V1

Supermon is a utility to aid the programmer with the development and testing of machine language programs. In order effectively to use this utility, knowledge of the 6502 assembly language is required. Several books have been published that describe the 6502 microprocessor and its instruction set. Use one of these books as your reference. Commodore sell an Assembler Tutor package to aid your learning. A complete Assembly Development Package is also available.

1. To load the program, enter LOAD"SUPERMON64.V1",8(R)
2. To run it enter RUN(R)

You have now entered the world of "HEXADECIMAL"; therefore, all addresses and values must be entered in this notation. Again, consult a manual for further explanation.

In order to distinguish the MONITOR's command mode from that of BASIC's, the full stop is used as the prompt. The commands that are valid to SUPERMON are:

L	Load next program from cassette
L"XXX"	Load program XXX from cassette
L"XXX",08	Load program XXX from disk
M 0000 0080	Display memory from 0000 to 0080
R	Display/edit contents of registers
:2000 FF	Change memory location 2000 to the hex value FF (you can change the 8 locations starting at 2000 by separating each value by a space)
H C000 D000 STRING	Hunt through memory for ASCII STRING
F 1000 1100 FF	Fill memory with hex value FF
T 1000 1100 5000	Transfer section of memory to 5000
A 2000 LDA # $12	Enter an assembler instruction at 2000
D 2000	Disassemble 22 instructions starting at 2000
P 2000 3000	Disassemble instructions starting at 2000 and ending 3000
S"XXX",01,0800,0901	Save code from 0800 t0 0900 as file XXX on device 01 (cassette)
G	Run code starting at the address pointed to by PC (Program Counter)
G 1000	Run machine-language code starting at hex 1000
X	Return to BASIC

To re-enter SUPERMON type SYS 38893.

NOTE
All commands must be followed by pressing the RETURN key. All values shown above are in hexadecimal notation.

2.8 PET EMULATOR (BASIC 2.0 OR LESS)

2.8.1 MEMORY CONFIGURATION

The COMMODORE 64 in normal operation, stores BASIC programs in the $0800 to $9FFF memory range (HEX) with the screen stored at $0400 to $0800. The PET stores BASIC programs at $0400 to $7FFF with the screen stored at $8000 to $8400.

The Emulator re-configures the COMMODORE 64 memory so that it duplicates the PET internally. Thus POKES to the screen, POKES to the program, and other such direct access operations work properly.

NOTE: If POKEs to the screen are to be visible add the following to the program whenever the screen is cleared.

```
POKE 53281,CC:PRINT"CLS":POKE532181,BC
```

where CC is the colour the poked characters are to be and BC is the normal colour of the screen.

2.8.2 SYSTEM INTERACTION INTERPRETATION

Many PET programs access the system directly with PEEKS, POKES and WAITS. Most of the common PEEKS, POKES and WAITS are interpreted by the Emulator and should operate exactly as they would on the PET. A few of these locations are as follows:

	PEEK	POKE	Operation
50003	x	x	BASIC version type test
59464	x	x	CB2 sound frequency
59467	x	x	CB2 sound on/off
59466	Set to 15		
59468	x	x	Set upper/lower case

All POKES and PEEKS between $0000 and $03FF (when possible) are translated. POKES not able to be interpreted return the message 'illegal quantity error'.

Cassette buffer #2 is also available for machine code programs (same as the PET). Machine language programs that do not call system routines will work with no modifications if they reside in this cassette buffer.

The CB2 sound is emulated as closely as possible. Certain very high tones available on the PET cannot be obtained on the 64, and the pitch of the tones varies across the scale. Musical tunes may not be emulated correctly, but other sound effects usually sound better under the emulator than they have on the PET 2.0 Operation.

2.8.3 LOADING THE PET EMULATOR

The Emulator loads into high memory on the COMMODORE 64, $C000 (HEX). It may be loaded directly by typing: LOAD" PET EMULATOR",8,1(R). (Note the space between the first quotation mark and the filename.) When the computer replies READY type SYS 12★4096. You are given the opportunity to select screen colour before the PET Emulator begins executing.

Once the Emulator is running, you need only LOAD and RUN your PET programs.

2.9 DOS WEDGE

Included on the diskette is another valuable product from Commodore, the DOS 5.1 Wedge program. This program is more commonly called the DOS Wedge because it "wedges" itself into the operating system and BASIC interpreter. Thus, the wedge checks all keyboard entries for its command characters before passing the entry on to the BASIC interpreter. (This is done by linking into the CHRGET Operating System routine in page zero.)

The DOS Wedge provides a very useful 'short cut' method for communicating with the disk drive. It cannot make the disk drive do anything more than can be done through commands in a BASIC program, but allows the user to communicate in a direct mode.

To load the wedge program, enter LOAD" DOS WEDGE",8,1. (Note the space between the first quotation mark and the file name.) This loads a program that "boots" the actual wedge program into memory. Once loaded, type: SYS 52224 RETURN. When the wedge program is activated, a copyright notice is displayed.

2.9.1 USING THE DOS WEDGE

The wedge program supports all of the same commands that are included in BASIC (copy, scratch, rename, new a disk) a command to read the directory (without overwriting memory) and commands to load and run programs. The wedge program also provides you with the capability of creating and maintaining volumes of files (volume creation allows you to group certain programs together) and the capability to perform operations using a wild card filename (any file whose name begins with certain characters).

Each command begins with a single character as specified in the following paragraphs. The character used depends on the command. The @ (commercial "at" sign) and GREATER THAN are used interchangeably to begin any of the disk housekeeping commands or to read the directory. They are also used to reset or initialise the drive, and to terminate the DOS Wedge. The UP ARROW is used to initiate the command to load (at BASIC's Start of Text address) and automatically run a program. The / (divide sign) is used to initiate the command to load a program at BASIC's Start of Text address. The % (percent sign) is used to initiate the command to load a program at its load address. Finally, the LEFT ARROW is used to initiate the command for saving a file to disk.

2.9.2 COMMAND DESCRIPTIONS

A description of each command is given in the following pages. Please note that parentheses specify an optional parameter.

@

Typing this character alone provides the user with the current disk status. This performs the same function as the following BASIC code:

```
10 OPEN 15,8,15
20 INPUT#15,A,B$,C,D
30 PRINT A;B$;C;D
```

@$(drive):(filename)(*)([volume])

This command reads the directory from the disk drive specified and prints it to the screen. If filename is specified, all files whose names begin with the letters specified by filename will be printed. (If [volume] is specified (where volume is the character id of that volume), then only those files contained on that volume will be printed.

@N(drive):diskname,id

This command formats a disk using the name and id specified.

@R(drive):newfile([volume]) = oldfile([volume])

This command renames the file specified by oldfile to the name specified by newfile.

@C(drive):newfile([volume]) = oldfile([volume])

This command copies the file specified by oldfile to the name specified by newfile. If volume is specified, the newfile is created on that volume.

@S(drive):filename(*)([volume])

This command scratches the files specified by filename. If * is specified, all files beginning with the letters specified by filenames are scratched. If volume is specified only those files that are contained on that volume are scratched.

@U:

This command resets the DOS.

@I(drive)

This command initializes the disk drive.

@Q

This command terminates the wedge program.

/filename

This command loads the file specified by filename. For example:

```
/ASM.C64
```

causes the program named "ASM.C64" to be loaded into memory. This command does the same thing as the BASIC command:

```
LOAD"ASM.C64",8
```

NOTE
This command can only be used to load BASIC programs, or machine code programs that are booted from BASIC. This is because the computer will ignore the file's own load address and will instead load at the current "Start of BASIC Text" area.

%filename

This command loads the file specified by filename at its own load address. It does the same thing as the BASIC command:

```
LOAD"filename",8,1
```

where filename is the name of the program to load.

↑filename

This command allows the user to load and run the program specified by filename and does the same thing as entering:

```
LOAD"filename",8
```

followed by the BASIC command RUN.

NOTE
This command can only be used to load and run BASIC programs, or machine code programs that are booted from BASIC.

←filename

This command saves to disk, the program specified by filename.

SECTION THREE GRAPHICS

3.1 CHARACTER EDITOR

The characters that are printed onto the screen through PRINT statements (or POKE statements to screen memory) are stored in ROM. In the Commodore 64, there are two such character sets, a GRAPHICS set (the set that is active when machine is switched on) and a LOWER CASE set. Each set occupies 2048 (2K) bytes. On the Commodore 64, it possible to program five alternate character sets.

The Character Editor allows you to create alternate character sets (for the screen) and store them on disk.

The following character sets can be set up in RAM.

Character Set	Enable With	loaction
1	POKE 53272,19	2048-4095
4	POKE 53272,25	8192-10239
5	POKE 53272,27	10240-12287
6	POKE 53272,29	12288-14335
7	POKE 53272,31	14336-16383

The built-in character sets can be selected as follows:

2(Graphics)	POKE 53272,21
3(Lower Case)	POKE 53272,23

To load this program enter LOAD"CHAR BOOT",8 and press RETURN. Once loaded, enter RUN and press RETURN.

3.1.1 CHARACTER SELECTION MODE

The program initially comes up in Character selection mode. In this mode, a box cursor flashes over one of the 64 characters displayed at the bottom of the screen. A character can be selected by moving the cursor over that character and pressing RETURN. You also can select another character set, change the background or border colour, or save a character set by using one of the following commands:

1	Select/display character set 1
2	Select/display character set 2 (cannot be edited)
3	Select/display character set 3 (cannot be edited)
4	Select/display character set 4
5	Select/display character set 5
6	Select/display character set 6
7	Select/display character set 7
CONTROL-N	Displays the next 64 characters of the current character set
CONTROL-B	Steps through each of the 16 background colours
CONTROL-E	Steps through each of the 16 border colours
CONTROL-L	Load a character set file from disk, e.g. " COMPUTER.SET 5" which is included with this package
S	Saves a character set in a named file on disk
Q	Quit (exit to BASIC command mode)

Once you have selected the character you wish to edit (by moving the cursor to that character and pressing RETURN), you will enter the Character edit mode. That character selected then appears in the EDIT GRID at the upper-left of the screen. The character is also displayed in all 16 colours to the right of the EDIT GRID.

3.1.2 CHARACTER EDIT MODE

The Character edit mode allows you to create, edit, and design single characters that can be stored in a character library. A character is made of dots. The following commands allow you to turn specific dots on and off in order to design a particular character:

FULL STOP	Turn dot under cursor on
SPACE BAR	Erase dot under cursor
DEL	Erase dot to left of cursor
CLR	Erase all dots on grid
CONTROL-R	Reverse all dots on grid

The following commands allow you to move the cursor within the box:

CURSOR RIGHT	cursor right
CURSOR LEFT	cursor left
CURSOR UP	cursor up
CURSOR DOWN	cursor down
HOME	cursor to top-left
RETURN	cursor to first position of next line

The following commands allow you to move a character around on the grid:

F1	Character up one line
F3	Character down one line
F5	Character left one line
F7	Character right one line
£	Rotate character 90 degrees

The following commands allow you to change colours (i.e., step through each of the 16 colours):

CONTROL-E	Change border colour (edge)
CONTROL-B	Change background colour

The following command returns you to Character selection mode:

Q	Return to Character Selection Mode

To assign Edited character to character set press CONTROL and A then position cursor and then press RETURN.

3.2 SPRITE EDITOR

The Sprite Editor allows you to create one or more sprites on the screen, dot by dot, on a 21 line by 24 column grid. A sprite is a special type of user-designed object which can be displayed anywhere on the screen. These sprites can be displayed in a fixed location or as moving objects. When the sprites you have created are acceptable, the editor allows the option for storage to a disk file for future use with BASIC or ASSEMBLER programs.

Each sprite definition uses 64 bytes of memory, called a "page". Graphically, a page is set up as a 3 by 21 byte grid with the 64th byte not used. You can have up to 160 sprite-definitions in memory at a time, as follows.

Sprite Page	Memory Locations	Number of Sprites
32-63	2048-4095	32
128-255	8192-16383	128

When all of your sprite editing is done and stored on a disk file, at some point you will want to recall the sprites into memory for usage with a BASIC program.

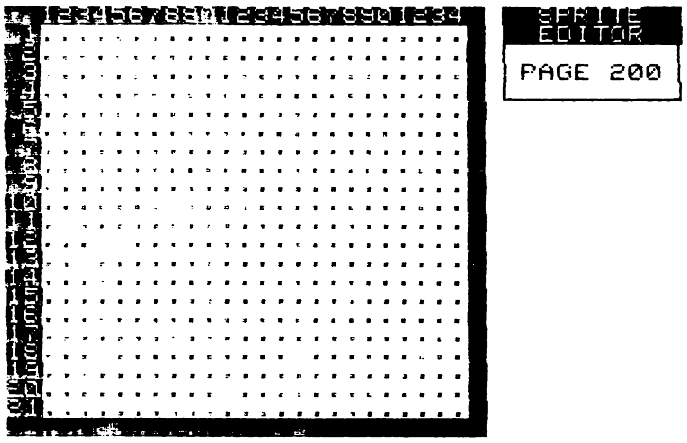
Due to the characteristics of BASIC, the following technique is required to correctly load the sprite information:

```
10 REM LOAD IN THE SPRITES
20 IF SW=1 THEN 50
30 SW=1
40 LOAD"SPRITEFILE",8,1
50 REM CONTINUE PROGRAM AFTER "LOAD"
```

To run the Sprite Editor type:

```
LOAD"SPRITE BOOT",8(R)
RUN(R)
```

The screen then looks like this:



SPRITE EDITOR PAGE 200

Note that the Sprite Editor always starts with sprite 200.

The following commands allow you to turn specific dots on and off in order to design a particular character:

FULL STOP	Turn dot under cursor on
SPACE BAR	Erase dot under cursor
DEL	Erase dot to left of cursor
CLR	Erase all dots on grid
CONTROL-R	Reverse all dots on grid

The following commands allow you to move the cursor:

CURSOR RIGHT	Cursor right
CURSOR LEFT	Cursor left
CURSOR UP	Cursor up
CURSOR DOWN	Cursor down
HOME	Cursor to top-left
RETURN	Cursor to beginning of next line

The following commands allow you to move a sprite around the grid:

F1	Move sprite up one line
F2	Move sprite down one line
F3	Move sprite left one column
F4	Move sprite right one column
ENGLISH POUND	Rotate sprite 90 degrees

The following commands allow you to select a different sprite to be edited.

+	Display next sprite (i.e. add one to page number)
–	Display previous sprite (i.e. subtract one from page number)
CONTROL-P	Select a specific page

The following commands allow you to change colours (i.e., step through each of sixteen colours):

>	Change colour of sprite
CONTROL-E	Change edge (border) colour
CONTROL-B	Change background colour

The following commands allow you to change the size of the current sprite:

CONTROL-X	Expand/contract horizontally
CONTROL-Y	Expand/contract vertically

The Sprite Editor starts up in HiRes mode. You can change mode with these commands:

M	Multi-colour mode
H	HiRes mode

The following commands allow you to display sprites (the current sprite is always displayed at the bottom-right of the editing screen):

CONTROL-D	Display a range of pages (8 per screen)
CONTROL-V	Clear screen and display current sprite in motion; you can then alter the motion of the sprite as follows:

+	Speed Up
–	Slow down
SPACE BAR	Stop/start
RETURN	Return to editing screen

The following commands allow you to load and save sprite definitions on disk:

S	Save a range of pages
CONTROL-L	Load a sprite-definition file such as " SAMPLE SPRITES" which is provided in this package.

This command returns to BASIC:

Q	Quit the program

SECTION FOUR SOUND

4.1 INTRODUCTION

SIDMON is a program which allows you to create sounds using the 6581 Sound Interface Device (SID) in the COMMODORE 64. Rather than randomly trying different sounds by poking around until you get the sound you want, SIDMON does all of the poking for you. You just look at the screen and change whatever you wish. If you do not like the change, change it back. It's that simple to use.

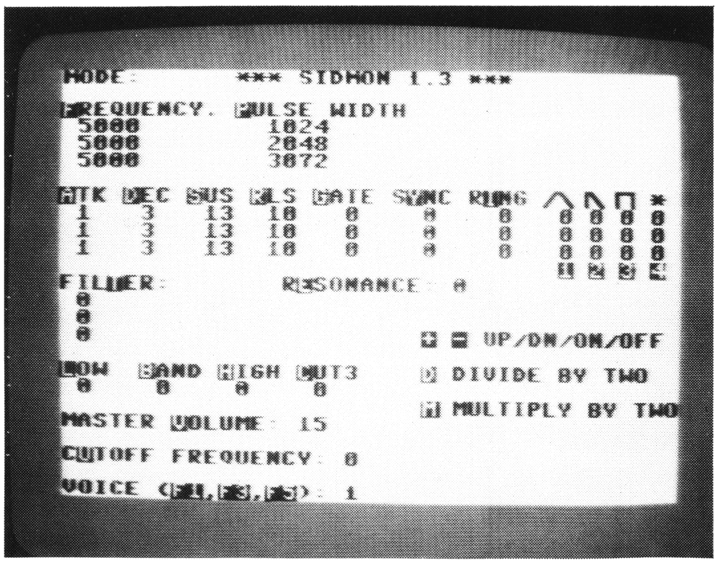
The commands are usually the first letter of the register name. In some cases though, the letter has already been taken by another command, so the second letter is used instead. The proper letter to press is highlighted in reverse on the screen. (See Figure **4**-1.)

4.2 LOADING SIDMON

To load SIDMON type:

```
LOAD"SIDMON",8(R)
RUN(R)
```

The following screen appears:



```
MODE:           *** SIDMON 1.3 ***

FREQUENCY.  PULSE WIDTH
 5000          1024
 5000          2048
 5000          3072

ATK DEC SUS RLS GATE SYNC RING  ^ ^ П *
 1   3   13  10  0    0    0    0 0 0 0
 1   3   13  10  0    0    0    0 0 0 0
 1   3   13  10  0    0    0    0 0 0 0
                                1 2 3 4
FILTER:         RESONANCE: 0
 0
 0
 0                        + - UP/DN/ON/OFF

LOW  BAND HIGH  OUT3      D DIVIDE BY TWO
 0    0    0    0
                          M MULTIPLY BY TWO
MASTER VOLUME: 15

CUTOFF FREQUENCY: 0

VOICE (F1,F3,F5): 1
```

FIGURE 4-1 SIDMON SCREEN

Commands are entered by pressing the key (shown in reverse) associated with that function. Up/down and on/off settings are controlled by pressing the + and - keys. Although more detail is provided in later sections, this is best described by using examples:

1. Select your waveform (triangular, sawtooth, square or pulse, white noise) by pressing 1, 2, 3 or 4

2. Gate the sound on by pressing G followed by +

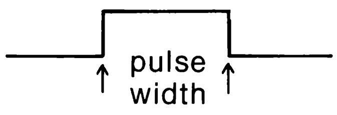
3. Alter the frequency by pressing F followed by:

   + to raise the frequency
   - to lower the frequency
   M to multiply the frequency by 2
   D to divide the frequency by 2

Attack decay, sustain, release etc. can be altered in the same manner. Any one of the three voices can be selected by F1, F3 or F5.

After creating the desired sound, jot down the numbers in the registers for later use in your program.

4.3 DESCRIPTIONS OF THE DIFFERENT REGISTERS

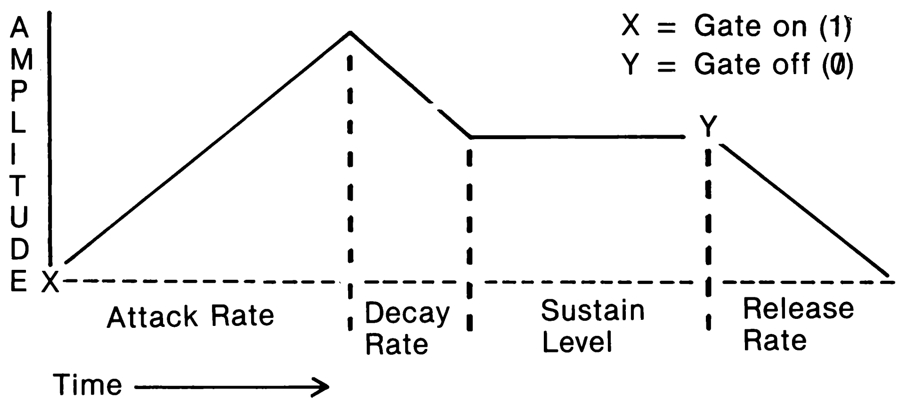
FREQUENCY	This register sets the number of sound waves per second produced by the SID.
PULSE WIDTH	This register forms a number which linearly controls the duty cycle of the pulse waveform on the voice that is being worked on. 
ATTACK	The attack rate determines how rapidly the output of the voice rises from zero to peak amplitude when the voice is gated.
DECAY	The decay cycle follows the attack cycle and the decay rate determines how rapidly the output falls from peak amplitude to the selected sustain level.

SUSTAIN

This selects one of 16 sustain levels for the voice. The sustain cycle follows the decay cycle and the voice remains at the selected sustain level as long as the gate bit remains on.

RELEASE

There are 0 to 15 release levels of which one can be selected. The release cycle follows the sustain cycle when the gate bit is turned off (reset to zero). At this time, the output of the voice being worked on falls from the sustain amplitude to zero amplitude at the selected release rate.



GATE

When the gate is set to 1, the attack/decay/sustain cycle begins. When the gate is reset to 0, the release cycle begins. The gate must be set for the selected output of the voice to be audible.

SYNC	The sync, when set to 1, synchronizes the fundamental frequency of the voice being worked on with the frequency of voice 3. Varying the frequency of the present voice with respect to voice 3 produces a wide range of harmonic structures from the present voice at the frequency of voice 3. In order for sync to occur, voice 3 must be set to some frequency other than zero but preferably lower than the frequency of the present voice. No other parameters of voice 3 have any effect on sync.
RING	The ring register, when set to 1, replaces the triangle waveform of the present voice with a ring modulated combination of the present voice and voice 3. Varying the frequency of the present voice with respect to voice 3 produces a bell or gong-like sound.
Λ	The triangle waveform has a mellow, flute-like quality.
◸	The sawtooth wave form is rich in even and odd harmonics and has a bright, brassy quality.
⊓	The harmonic quality of the pulse waveform can be adjusted by the pulse width register, producing tone qualities from a bright, hollow square wave to a nasal, reedy pulse.
*	This output is a random signal. The sound quality can be varied from a low rumbling to a hissing white noise via the frequency register.
FILTER	When set to 0, the present voice appears directly at the audio output and the filter has no effect on the sound. If set to 1, the present voice is processed through the filter and the harmonic content of the voice is altered.
RESONANCE	This register controls the resonance of the filter. Resonance is a peaking effect which emphasises frequency components at the cutoff frequency of the filter, causing a sharper sound. There are 16 resonance settings ranging linearly from no resonance (0) to maximum resonance (15).
LOW	When set to 1, the low pass output of the filter is selected and sent to the audio output. For a given filter input signal, all frequency components above the cutoff are attenuated at a rate of 12dB/octave. The low pass mode produces full bodied sounds.

BAND	This is similar to low, but all frequency components above and below the cutoff are attenuated at a rate of 6dB/octave. The band pass mode produces thin, open sounds.
HIGH	This is the opposite of low, in that when set to 1, all frequency components below the cutoff are attenuated at a rate of 12dB/octave. This mode produces a tinny buzzy sound.
CUT 3	When set to 1, the output of voice 3 is disconnected from the direct audio path. Setting voice 3 to bypass the filter and setting cutoff 3 to 1, prevents voice 3 from reaching the audio output. This allows voice 3 to be used for modulation purposes without any undesirable output.
MASTER VOLUME	Select one of 16 overall volume levels for the final composite audio output. The values this register can hold range from 0 (no sound) to 15 (max sound).
CUTOFF FREQUENCY	This register is set to contain a number which controls the cutoff (or centre) frequency of the programmable filter.

4.4 SIDMON COMMANDS

There are several commands that are used in SIDMON. The letters to initiate these commands, are highlighted on the screen in reverse. When a valid key is pressed, that key is displayed next to "MODE" on the top line. The only keys not displayed are +, -, M and D. For these keys to have any effect, another letter must be already displayed next to the mode.

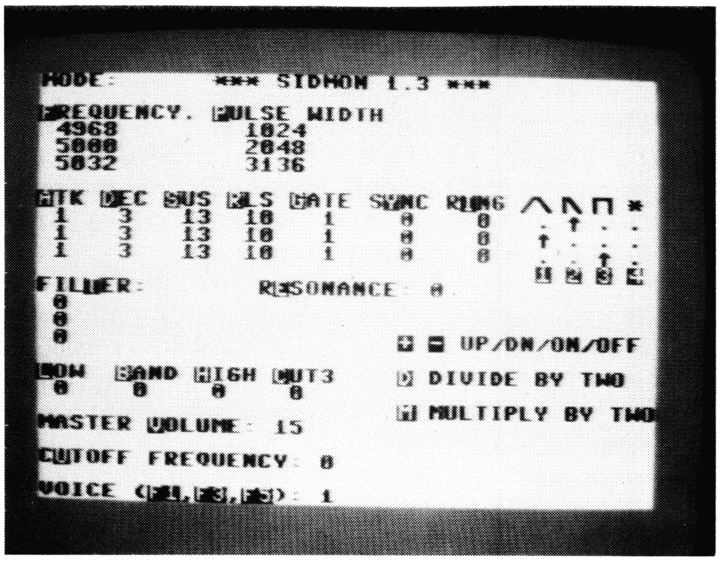
CLR/HOME	To completely reset SIDMON, press the SHIFT and CLR/HOME. All of the values are reset to their default.

WARNING
ANY CHANGES YOU HAVE MADE WILL BE LOST.

F1, F3 and F5	These three function keys are used to select the voice on which you are currently working. F1 is voice 1, F3 is voice 2 and F5 is voice 3. The voice that is presently being worked on is noted next to "VOICE (F1,F3,F5):" on the bottom line of the screen.
+/-	The add and subtract keys are used to raise or lower values in the different registers. In some cases, such as "gate" which is either on or off, the + key is used to turn on and the - key is used to turn off.

M/D	The M key and the D key are used to multiply and divide respectively. The keys work only on frequency and pulse width.
F	The F key is used to select frequency. Follow this key with either +, -, M or D to change the frequency. While in frequency mode, the + and - keys add or subtract 16 from the frequency of the voice on which you are working. If you only want to add 8 to the frequency multiply the frequency by 2 by using the M key, then using the + key add 16, then divide by 2 using D. Your frequency should now be increased by 8.
P	To select pulse width press the P key. The pulse width mode is similar to the frequency mode. The only difference is that the + and - keys change the pulse width by 64. Remember pulse width only applies when you are using square wave.
A, D, S and R	These keys select attack, decay, sustain and release respectively. The values of each range from 0 to 15. Plus and minus change these registers by 1. The M and D keys do not work while in this mode.
G	This key selects gate. A 1 (+) in this register starts the attack, decay and sustain. When reset to 0 (-), the release sequence begins. This register must be manipulated for you to get any sound from the SID.
Y and I	These keys select sync and ring. On and off are the only 2 states these registers have. The + key turns the register on and the - key turns it off.
1, 2, 3 and 4	Keys 1 through 4 select the type of waveform of the voice you are working on. The 1 key selects triangle, 2 sawtooth, 3 pulse and 4 noise. For pulse width to have any effect, the pulse waveform must be selected. Only one of these four choices may be on at any one time.
T	This selects filter. As in gate, sync and ring, the filter has either an on or an off state.
E	Resonance can be set by pressing the E key. It has a range of 0 to 15 which can be modified by using the + or - keys.
V	SIDMON has a master volume which can be changed by pressing V followed by + or - key.
U	The cutoff frequency is selected by pressing U. This, followed by + or -, changes the cutoff frequency. D or M do not work in this mode.

4.5 SAMPLE SOUNDS



```
MODE:          *** SIDMON 1.3 ***

FREQUENCY.  PULSE WIDTH
 4968          1024
 5000          2048
 5032          3136

ATK DEC SUS RLS GATE SYNC RING  ^ ^ ⊓ *
 1    3  13  10   1    0    0   . ↑ . .
 1    3  13  10   1    0    0   ↑ . . .
 1    3  13  10   1    0    0   . . ↑ .

FILTER:         RESONANCE: 0
 0
 0
 0                         + - UP/DN/ON/OFF

LOW  BAND HIGH OUT3        / DIVIDE BY TWO
 0    0    0    0
                           * MULTIPLY BY TWO
MASTER VOLUME: 15

CUTOFF FREQUENCY: 0

VOICE (F1,F3,F5): 1
```

FIGURE 4-2 SAMPLE SOUND SETUP

This produces a harmonious sound using offsets of one major frequency (voice2).

For more information on the SID see the Programmer's Reference Guide.

SECTION FIVE BASIC PROGRAMMING AIDS

5.1 INTRODUCTION

The COMMODORE 64 DATA SCREEN EDITOR allows you to create BASIC programs with the aid of a Screen Field Editor. It provides the user with a very powerful software tool that allows the definition and editing of data fields that are input through the screen display. With the COMMODORE 64 SCREEN EDITOR, screen design and the input and editing of data which have always been one of the more difficult tasks facing the BASIC programmer is not only easier, but accomplished with the accuracy and speed of Assembly Language programming. The SCREEN EDITOR program, written in the 6502/6510 language, also gives the user several additional BASIC commands. This program contains everything that the user needs to create, edit, and manipulate data fields from within a BASIC program. This section is directed towards the experienced computer user who already has some familiarity with the BASIC language and the operations of the COMMODORE 64 computer. The program is not intended to provide the knowledge of 'how to' in BASIC language, but provides the software tools for the experienced programmer to become even better.

To load the screen editor, type:

```
LOAD"editor64",8,1(R)
SYS 49152(R)
```

Press RETURN when the statement LOAD":*",8 appears. This loads a demonstration program. (The same procedure can be used to load and run your own program, substituting the program name for *.)

5.2 USER CONVENTIONS

A brief description of certain keys and symbols, and their respective function in reference to the SCREEN EDITOR program follows.

Comma and Full Stop	For editing numeric 'dollar' amount fields.
Left and Right Parentheses	Used to define a NUMERIC ONLY field.
Left and Right Bracket	To define an alpha-numeric field.
Greater than and Less than	Used to define an alphanumeric field.
(R)	Continues with program after input of line.

5.3 DESCRIPTIONS

5.3.1 FIELD DESCRIPTIONS

The COMMODORE 64 SCREEN EDITOR uses three 'types' of fields for the purpose of data entry. These fields are:

1. Alphanumeric, non-field edit, left to right entry
2. Alphanumeric, field edit, left to right entry
3. Numeric, right to left entry

Each of the three data types are defined on the screen as:

[ ]	Names, addresses, cities, etc. Legal data for this field is any character from the keyboard.
/ /	Dates, part numbers, telephone numbers, etc. Legal data for this field is any character from the keyboard except the comma (,), period (.), slash (/), and dash (-) which are simply passed over for easy editing by the Screen Editor program.
(,)	Monetary amounts, quantities, etc. Any number is legal in this field. Commas and decimal points are passed over for easy editing by the Screen Editor program.

ALL of these data fields can be placed at any location of the screen except the first and last lines of the screen. The first and last lines are reserved as Program Status Lines.

NOTE

A negative (dash sign) before a field, or credit flag ('cr') after a field, changes the field sign To do this POKE 52921,x where x is 0 for the dash sign and not 0 for the cr.

5.3.2 COMMAND DESCRIPTIONS

The following commands have been added to the BASIC command list. A description of each follows.

The Screen Editor program requires that the first character of these commands be the '&' symbol.

&b To display the bottom Program Status Line

&c To change colour combinations

&e To enter the data entry/edit mode, allowing the user to edit any field on the screen

&f To edit a single field on the screen

&k To enable the use of the RUN/STOP key

&l	To draw a horizontal line across the screen
&s	To disable the use of the RUN/STOP key
&t	To display the top program status line

5.3.2.1 Command suffixes for field editing (&e and &f)

Two of the above commands, '&e' and '&f', have optional suffixes to make the command more versatile. For example:

The command '&e,(5)' places the cursor at the beginning of the fifth field on the screen. You may then edit as many fields as you desire. The range of the field selection is 0 to 99.

The command '&f,(12)' places the cursor in the twelfth field. The twelfth field can then be edited. Control is then returned to the BASIC program.

If the field specified with '&e' and '&f' does not exist, the program prints the following message:

```
?undef'd field error
ready.
```

5.3.2.2 Parameters of colour command (&c)

To change the colour of the background, border, and characters, use the '&c' command. For example: &c,"XYZ" specifies that the character colour is to be 'X', the border colour is to be 'Y', and the screen colour is to 'Z', where, 'XYZ' are the actual control characters of the colours you desire. (These characters appear 'reversed' in the command display.) The default colours are black, black, and white for the border, background, and character colours respectively. This command must have all characters specified, or the value not specified will be replaced by the default colour.

5.3.2.3 Screen printing commands (&l)

The commands '&l' is for screen printing, i.e. for separating data.

This command directs the program to print a line of 40 characters across the screen at the next available print position. The default character to be printed is ' = '. To change this character, simply POKE the address with the ASCII value of the character you wish to have printed.

For example, to produce a line of dashes (-):

```
POKE 49360,ASC("-"):&l
```

5.3.2.4 Program Status Line commands (&t and &b)

The Screen Editor displays two bars referred to as the Program Status Lines. These lines are the top and bottom lines of the screen. The next two print commands are specifically for each of the Program Status Lines:

&t The '&t' command displays the top program status line, erasing the previous contents.

&b The '&b' command displays the bottom program status line, replacing the previous contents with "busy".

With either the '&t' or '&b' command, the Screen Editor program automatically displays 'busy' on the bottom status line. This allows the programmer to print "busy" without having to code another BASIC subroutine.

5.3.2.5 Stop key commands (&s and &k)

The last two commands are concerned with the operation of the STOP key.

&s To disable the STOP key

&k To enable the STOP key

NOTE
When the STOP key is disabled, the internal clock continues to function normally.

5.3.3 OPERATION DESCRIPTIONS

It is relatively easy to operate the Commodore 64 Screen Editor. Simply type the data into the fields via the computer keyboard. Any typed character that does not comply with the rules of each particular field is ignored. The computer operator may also use the standard cursor control keys. The operation descriptions for the standard cursor keys are detailed below:

CLR	The CLR key clears all fields on the screen and places the cursor at its default location.
CRSR DOWN	The CRSR DOWN key (unshifted vertical arrows) moves the cursor to the beginning of the next field. If there happens to be no other field, the Screen Editor program automatically goes into the Screen Accept Mode.
CRSR LEFT	The SHIFT key in conjunction with the CRSR key with the horizontal arrows moves the cursor one position to the left.
CRSR RIGHT	The CRSR RIGHT key (unshifted horizontal CRSR key) moves the cursor one position to the right.

CRSR UP	The SHIFT key in conjunction with the vertical CRSR key moves the cursor to the beginning of the previous field. If there doesn't happen to be a previous field, the command is ignored.
DEL	The DEL key deletes the previous character entered in the field and moves all data after the cursor, one position to the left.
HOME	The HOME key places the cursor at the beginning of the first field with no data loss.
INST	The SHIFT and INST/DEL key inserts a space into a field and moves all data after the cursor one position to the right.
UP ARROW	The UP ARROW key ↑ deletes the character under the cursor and shifts the data after the cursor one position to the left.
- (minus)	When used in a numeric field sets or resets the negative (-/cr) flag.

These keys operate in any field except the numeric fields. The numeric fields cannot accept the INST key or the UP ARROW key because the data in the numeric fields is right justified. Therefore, the INST and UP ARROW keys become unnecessary.

5.4 OPERATIONS

5.4.1 CONTROL OPERATION

The Commodore 64 Screen Editor program has a built in 'control mode' routine. This routine can be modified by the programmer to produce status codes on any key on the keyboard. The 'defined' codes are limited to four.

To use the control mode, simply press the control (CTRL) key. The program then displays 'control mode' on the bottom Program Status Line. To exit back into the main program, simply press the 'Commodore' key.

The control mode has two preset options. These options are 'e' and SHIFTed 'q'. The 'e' key exits from the basic program and loads the first program found on the disk drive. A RUN is also pushed into the keyboard buffer to assure the execution of the program.

WARNING
THE SHIFTED Q COMMAND SHOULD BE USED WITH EXTREME CAUTION BECAUSE IT RESETS THE COMMODORE 64, i.e., REMOVES THE BASIC SCREEN EDITOR PROGRAM FROM MEMORY.

For the four keys a programmer can define, use the following table to learn the ASCII values which can be POKEd into the program much like the line command. For example, to produce an ST 02 with the F1 key, input 'POKE 51350,ASC("F1")'. The status, 'ST', value is the value which is returned to the BASIC program by the Screen Editor.

POKE	ST
51350	02
51361	03
51372	04
51383	05

The ST is used so that the BASIC programmer has easy access to the status of the Screen Editor.

5.4.2 ARRAY OPERATION

The Screen Editor stores the contents of the screen fields in the string array 'SC$'. This enables the BASIC programmer to process the data returned from the editor. The editor fills the array with as much data as possible. The programmer MUST define the array and fill the array with spaces to match the field lengths.

The first field on the screen is stored in array element zero (0). The second field is in location one. This process continues for each field until the array is full.

5.4.3 COMMANDS WITHIN PROGRAMS

When entering extra commands within a BASIC program, enter a colon first, e.g.:

```
10 :&t
20 IF A = 0 THEN :&b
```

If the colons are omitted, in this example, line 10 would be executed on pressing RETURN and in line 20 &b would be executed every time as if it were in front of IF.

```
DIRECTORY OF FILES


0  "c64 utilities [illegible]
11     "Editor64 Demo"       prg
8      " c64 menu"           prg
4      "change disk"         prg
10     "copy-all64"          prg
28     "1541 backup"         prg
2      "dump"                prg
2      "load addr"           prg
10     "supermon64.v1"       prg
17     " pet emulator"       prg
4      " dos wedge"          prg
4      "char boot"           prg
32     " char editor"        prg
1      " rotate.data"        prg
9      " standard.set"       prg
9      " computer.set 5"     prg
3      "sprite boot"         prg
43     " sprite editor"      prg
2      " scroll.data"        prg
3      " sample sprites"     prg
18     "sidmon"              prg
16     "editor64"            prg
1      "   directory    "    seq
1      "dos boot"            prg
1      "emu boot"            prg
425 blocks free.


FOR SECURITY PLEASE BACKUP THIS DISK

USE "1541 BACKUP".
```

COPYRIGHT

COMMODORE 64 SOFTWARE

Commodore Business Machines (UK) Ltd
675 Ajax Avenue, Slough
Berks, SL1 4BG.

Made in England.

commodore COMPUTER