SUCCESS for the Sinclair ${\tt QL}$

CONTENTS

1.	INTRODUCTION
1.1	Why an Emulator?
1.2	Getting Started
1.3	How to Succeed with SUCCESS
2.	RUNNING CP/M SOFTWARE
3.	SUCCESS AT THE CP/M COMMAND LEVEL
3.1	Filenames
3.2	Peripheral Devices
3.3	Control Codes
3.3.1	Used on input
3.3.2	Used on output
3.4	Operating System Commands
3.5	Standard .COM-File (Transient) Commands
3.6	Special SUCCESS Commands
3.7	Terminal Emulation
4.	SUCCESS AT THE MACHINE-CODE LEVEL
4.1	How SUCCESS is Organised
4.2	BIOS
4.2.1	Resources
4.2.2	
4.2.3	BIOS Jump Block Extensions for SUCCESS
4.2.4	Special Extension: DEFKEY
4.2.5	Defining Drive G:
13	CP/M-Z80 Disk Structure



SUCCESS

THE SUPREME CP/M CODE EMULATION SYSTEM

by Brian Watson

Dr. HELMUT AIGNER & Dr. ANDY AIGNER



1. INTRODUCTION

1.1 Why an Emulator?

Whoever wishes to utilise any of the thousands of programs written for the CP/M-80 operating system on the Sinclair QL or CST Thor could either get a third processor - a very expensive option - or, preferably, obtain a Z80 emulator (Z80 being the name of the microprocessor for which CP/M was written). The latter option, though somewhat slower, is reasonably cheap. CP/M, by the way, stands for Control System for Microcomputers; it was developed by Digital Research.

An emulator is a program which will allow something to run on a machine which would normally only work on another machine. SUCCESS (SUpreme Cp/m Code Emulation SyStem) is an emulator for the Sinclair QL that will work in a CP/M (version 2.2) environment. With some extra programming it could, however, be used to emulate any Z80 environment (a Spectrum emulation, argghhhh!). SUCCESS will handle CP/M programs up to a length of 64 Kbytes. (This restriction may be lifted in a later version.) It will only run on a disk-based system.

The program works by taking each Z80 instruction from memory and then executing a 68000 machine-code subroutine depending on the instruction value. Being written in machine code, it is quite fast, permitting an emulation speed of roughly 1.0 MHz on a standard QL (about 1.8 MHz when on EPROM or in expansion memory).

1.2 Getting Started

Do curb your impatience to get SUCCESS up and running right away. Disks have been known to become faulty, and files to be erased by mistake. Murphy's Law decidedly states that it will happen to YOU if you fail to make a backup copy of your SUCCESS disk.

To make a backup copy, type

LRUN flpl clone bas

Following the program prompts, put the master disk in drive 1 and examine your blank disk, checking whether it is write-protected; if necessary, unprotect it. Now press Enter. Once all the essential files are copied, you will be asked whether the file named cpmfiles is to be copied, too. This is merely to reassure you in case you are getting impatient while 1042 sectors are being copied. We suggest you reply 'Y' to the prompt regardless: if you are a novice at CP/M, you can't know yet whether you will need cpmfiles; and if you're not, you'll need it for sure!

When the drives have stopped whirring and the lights gone out, remove your master disk and stow it in a safe place. Only the cloned disk should be used henceforth (except for making new clones in case something happens to the first one). The working copy you have just made is intended to work from drive 1; so put it there now before continuing.

Before continuing with SUCCESS, we strongly recommend that you load the file updates_doc into QUILL and peruse it for stop-press info on improvements not contained in the manual.

If you wish to go into Z80 assembler language, one of many excellent tutorials is:

> Alan Tully Z80 Reference Guide Melbourne House

There are also a number of fine books on 280 machine code for the ZX Spectrum.

You may further wish to contact a CP/M user group in order to obtain CP/M software, much of which is in the public domain, as well as for advice. The biggest of these in the UK is:

> PD Software 90 Braybourne Close Uxbridge, Middx, UB8 1UJ Tel: 01 8642633 or 0895 51978

Don't forget to specify the size of disk appropriate to your disk drive!

Other sources of CP/M software include:

Grey Matter 4 Prigg Meadow Ashburton, Devon, TQ13 7DF Tel: 0364 53499

Seltec Farley Hall, Wokingham Rd Bracknell, Berks, RG12 5EU Tel: 0344 863020

Shareware 87 High Št Tonbridge, Kent Tel: 0732 771344

OR BUY A COPY OF PERSONAL COMPUTER WORLD OR PRACTICAL COMPUTING: YOU'LL FIND LOTS OF ALTERNATIVE SOURCES.

MAWAWA 水中 P Classic 大 減別為別為另

TURBO V2.0

COMPATIBLE

THRMO AND PURBO TOOLKEY 92.0 - FASTER AND MORE CONTREVE THAN EVER RECORDED

Tubed AND 19880 TOOKETY P2.0 - FASTER AND SMEE CONFIGURE 1984 EVER BECKNETT

P2.0 (5 the ultimate vertice of the TUBED STRIPE, incorporating all the features purchase been mitting for, been before has the system been mitting for, been before has the system been mitting for the property of the system been mitting for the property of the system been mitting of the strong-best and religious purposes. The control of the strong-best and religious purposes.

Code address that (Librit or 120:1) - Object size (1484, 6485) - Top instancing BRIET, FAST or REPTS of leastly suitingation expenses the strong-best and the strong-best and the strong-best and the strong-best and the strong-best size of the str

TASE NAME (At Opposite to Intername) receives sections to the incention of DIPPO TZ, 01 to Apphalish by the Lack cord in preference and make at one incention and epaced and another properties of the INTRO CONTROL of the INTRO CONTROL of the INTRO CONTROL of the INTRO CONTROL OF TABLE 1, 11 to 10 to

Below we prevent the PCW brechmarks for TUPRO 92.0, with square format left on default GPRECOWN; imaging this to SIMUCIONID makes (5 on even faster) stronger prespond. (COPPULT MORAL) SCHOOL COPPULT NAMES (5 of 6) while makes (1 of 6) and makes (1 of 6) and makes (1 of 6) and (

Suprime (p. Smite m) TDBBO has you that so febberon switching diagnosts a ment full # all often mobile and femous menting influential in to global appending about for a statement block house) # size * size with the callect append in none, a heat of 15 organism. None in the divining south light was not been as report that principle countries are report that principle in south in the callect appendix in the area from the chance of 15 organism. So there follow the chance of 15 organism is a report that principle in the chance follow the chance of 15 organism is a frequent for 15 organism in the chance follow the chance of 15 organism is a strategie post feature fundament, and the chance following the chance of 15 organism is a strategie post feature fundament, and the chance of 15 organism is provided to 15 organism in the strategie of 15 organism is provided. Suprimedy post of 15 organism is provided, and the ONE RUNDING AND EACH of 15 organism is the chance of 15 organism is strategied to 15 organism is the first feet of 15 organism is the chance of 15 organism is the 15 organism is the 15 organism in 15 organism in 15 organism in 15 organism is the 15 organism in 15 o

possible

Supremely dusponenced – a THREE HUNDRED AND PIETY A4 PAGE, burd,
illustrated user mass (openia

Supremely grown as – there are no mywhy restrictions on TURBO'd programs.

The TURBO System costs £99.95 COMPLETE TARBO IS BRILLIANT PRESS COMPILET FROM EVERY SOURCE DELITERS IN AS THE MOST AUTOSTANDING RASIC COMPILET FOR ANY MICROCOMPILET.

TO TREBO THAT SHOULD RESERVE A STATE OF THE ST

Aram Ingangyas On brog	Donate of the control of	Old . Of Ment Desire	Net Pipes
E(#50# 15	& SCALE	RELEASE SASE	170(D DB
	POS CIPM		'n DLMT
	pert script	READ MEN	A_DINIA
	terul(IIIs	PC 53TE	4,0145
	THE EMPHYSIA	ROTATE L	# BIM7\$
	HOTEL PRINTS	POINTS T	¥"DIM75
	THPUS	SE PROFILE	K-SIM3
PERSON SIATUS	INTEREPS	SCREW ACTIVITY	e Diets
DAINE BANKINGS		SC COOLUT	a piwis
(p)1) T		STARCH MINORY	L_ELIEMD
47143			t Barra in
			1 (016 # 10
		SCI TONI	1,1(4)
	1151, 142	SEL DE LIBORAR	7 744011
		SEL POSITION	A MINUS
		SE1'PAIDARTS	A 34 +054
			A_ktensa
			¥7ri ≥o i
			¥.3€401\$
			A_D EMUSE
		14 # 1 # 5 9	YSTH
		STRIP COLOURS	4.22.112
			L24.15
* I=C .	STALE COLUMNS		¥_S1 T2\$
			# <u>"\$4724</u>
	Pills F T	1256 700918	M. BERD
		IASA PRIORITE	F 36136
			131131
		THISK TIMES	Magni Lingting.
	PLAS TIME	tentrii Ayery	MICL & PURKET
		I I Ar Ptolum	HATTE TLE
			MATE VE
			I HINE
	PARCITION		1 20 15 18
Francis To the Bright in the	PROG \$1.FE		1 31/11
St ti _			TEA OR MAT
	PENDENT!		6 MJAP.
	PCAD DH	Malniet 2156	L Dale In
CI IPPM	deft B) mr (TICES OF F	("Slift
		Control Cont	Part Part

UTSIARE BETT SUTER-MARGE BE THE REPORT SISSEM JUST 165 FOR THE BEST BE PARTE TO BEAREDS FROM THES EXCELERAL OFFER, REFLAM SOFFERMANDE MANUAL-LEWSLOW-MC

3. SUCCESS AT THE CP/M COMMAND LEVEL

3.1 Filenames

A CP/M filename consists of 3 parts: <drive>:<name>.<extension>

where <drive>: stands for one of the letters A to G followed by a colon

Example: A:PIP.COM

Drive assignment is as follows:

```
Drive A - FLP1
                 - 512K capacity, 128 files
Drive B - FLP2
                 - 512K capacity, 128 files
                 - 100K capacity, 64 files
Drive C - MDV1
Drive D - MDV2
                 - 100K capacity,
                                   64 files
                 - 200K capacity,
                                   96 files
Drive E - RAM1
                 - 200K capacity, 96 files
Drive F - RAM2_
                 - Format user-definable
Drive G - FLP2
```

Drives E and F must previously have been formatted at the SuperBASIC level. Drive G must have been established with the DEFINE command (see sections 2.1 and 4.2.5). Drive organisation can be altered by running the SuperBASIC program opmconfig bas.

The <drive>: part may be omitted from a filename, in which case it defaults to the current drive. On start-up A: is made the current drive; this may be changed by entering B: or C: etc.

The <name> part of a filename may consist of between 1 and 8 characters. All characters except space and <>.,::=?*[]_ are permitted (note that underline is illegal, as opposed to QDOS rules), and lower-case letters are automatically converted to upper case on input (this is true throughout CP/M).

The .<extension> part of a filename consists of a dot (full stop, period) and up to three characters as above. It may be omitted, in which case it defaults to an empty string. Commonly used extensions are:

```
ASM - Assembler source file

BAK - backup file

COM - command file

DAT - general data file

DOC - document file

DRV - raw-disk definition file

HEX - Intel Hex object file

KEY - function-key definition file

LIB - library file

MAC - Macro Assembler source file

PRN - Assembler list file

SUB - task submit file

TXT - text file

$$$$$ - temporary file
```

The use of wildcard filenames is permissible in certain commands. A wildcard filename differs from a simple filename in that both the name and the extension parts the characters? and * are also permitted, e.g. B:MAIL?.* (covering B:MAILO.DAT , B:MAIL4.DOC , etc; but not, say, B:MAIL15.TXT).

3.4 Operating System Commands

DTR

DIR <drive>:

DIR <filename>

DIR <wildcard> Lists non-system files

ERA <filename>

ERA <wildcard> Deletes files; stops at a write-protected file

REN <newfilename>=<oldfilename>

REN <newfilename> <oldfilename>

Renames a file (unless write-protected) Note that the new name is given first

SAVE <pages> <filename>

Opens a new file <filename> and stores in it <pages> * 256 bytes, starting from the initial address of the Transient Program Area

TYPE (filename) Copies a file to screen

USER <number> Switches to a different user area (0 on start-up) Range for <number> is 0 to 15

3.5 Standard .COM-File (Transient) Commands

ASM <filename>

Assembles an 8080 machine-code file. The file has to be created using ED (or any other text file editor). It will produce two files, <filename>.PRN - Printer output and <filename>.HEX. The .HEX file must be put through the LOAD program to produce a .COM file.

DDT Does not work; use DDTZ

DDTZ <filename>

This is a Dynamic Debugging Tool for Z80's. It is a piece of public-domain software which helps in program debugging. It is better than the (not included) standard DDT. The following sub-commands are recognised by the DDTZ program (command lines must not have more than 32 characters, or numeric arguments more than 4 digits):

A<address> ('assemble')

Stores command mnemonics (with hex arguments) starting at <address>

D<start>,<end> ('dump')

Lists memory between <start> and <end> addresses in both hex and ASCII format

F<start>,<end>,<value> ('fill')

Fills memory between <start> and <end> addresses with <value>

G<start>,<endl>,<end2> ('go')

Runs program, starting at address <start> until the command at either <endl> or <end2> is reached

```
Deletes n characters
      +/-nD
                Deletes n lines
      +/-nK
      nS<oldtext>^Z<newtext>^Z
                Removes <oldtext>, inserting <newtext> instead
      J<searchstring1>2<newtext>2<searchstring2>2
                Removes text between <searchstringl> and
                <searchstring2>, inserting <newtext> instead
                Ends editing session, makes changes permanent
      Е
                Ends current session and reopens file for editing
      Н
                Returns to state of last ED call of file
      O
                Ends editing session, abandons changes
      nM<commands> Executes <commands> n times
                Shows buffer contents
      OV
                Toggles line numbering on/off
      +/-٧
                Toggles upper-case conversion of input on/off
      +/-0
                Pause for n fiftieths of a second
      <filename>
LOAD
                Converts a file from Intel Hex format into a binary
                 .COM file. This can subsequently be run under {\tt CP/M}
                by entering the <name> part of <filename>
                           ('peripheral interchange processor')
PIP
       <destination file>=<source list>
PIP
       <destination file> <source list>
PIP
       <drive>:=<wildcard>
PIP
       <drive>: <wildcard>
PIP
       <filename>=<drive>:
PIP
       <filename> <drive>:
PIP
                 Copies information from one peripheral device to
                 another. Note that the destination is always given
                 first. <source list> may be one or more devices
                 and/or filenames (comma-separated). If no argument
                 is given, an asterisk prompt will be given at the
                 start of each line, and arguments as above can then
                 be entered line by line. Pressing ENTER will quit
                 this mode.
                 PIP is often used to copy files from one disk
                 drive to another; e.g. when a raw disk is defined,
                 PIP A:=G:*.*
                 copies all the files to the correct-format disks,
                 or
                 PIP LST:=PROGRAM.PRN
```

produces a printed copy of PROGRAM.PRN

ANALYSE

This utility, written in 68000 code, is to be used in conjunction with the DEFINE command (see below) and raw-disk read-write routines. It will attempt to extract as many details about a disk format as possible, starting with the disk sector size and density. This takes APSOLUTELY AGES (there are lots of possibilities to be exhausted) so don't jump to conclusions about a machine crash. When ready with this part, the routine will ask you several questions. If you are not sure of some of the parameters, guess. The utility will create a QDOS file named FLP1 DISKFMT DRV, which you must copy to the CP/M file, using the RDQDOS utility (see below). Note that ANALYSE will not be able to fully recognise some of the more obscure CP/M format variants: if it can't, it will fill in whatever details/parameters it can and will leave the rest for you to manually edit-in.

CAT

CAT <devname>

Displays the directory of a QDOS device, e.g. CAT FLP2_
The default is FLP1

DEFINE <definefile>

Defines the raw-disk drive, G:. The file specified contains information about the disk that is to be used. The default extension is '.DRV'. A few files for the more popular Z80 disk types will be found on the disk, viz.

AMS-DATA.DRV - Amstrad data-format disks

AMS-SYST.DRV - Amstrad system-format disks

BBCZ80.DRV - BBC micro Z80 2nd processor disks

These files show the exact format that the data should be in. When defining a .DRV file for a new type of drive, refer to the above versions. They all assume that the relevant size of disk drive is connected to (and defined as) FLP2_.

SETBAUD <baudrate>

Selects the serial-port transmission and reception rate, which should be one of: 150, 300, 600, 1200, 2400, 4800, 9600, 19200 (xmit only). If an invalid baud rate is typed, an error is given and the baud rate left as it was.

WRQDOS <cpmname> <qdosname>

Copies a CP/M-format file to a QDOS-format file. If the CP/M file does not exist or the QDOS file does, the copy will be aborted.

e.g.: WRQDOS BOOT.680 FLP1 BOOT

3.7 Terminal Emulation

Most CP/M programs are designed to be used on a system which emulates a terminal, most commonly a VT52. SUCCESS supports most VT52 emulation sequences, viz:

[ESC]A - Moves cursor up one line

[ESC]B - Moves cursor down one line

[ESC]C - Moves cursor right one character

[ESC]D - Moves cursor left one character

[ESC]E - Clears entire screen

[ESC]H - Homes cursor

[ESC]J - Erases to end of screen, including current line

[ESC]K - Erases to end of line

[ESC] Yrc - Moves cursor to (r-32), (c-32)

[ESC]p - Inverse video

[ESC]q - Normal video

PRECISION

Because &

3D PRECISION is a very classy 3-D system with SIERED ("Swift Translation and Elegant Rotation of Elements and Objects"). It is two programs in one. A full-feature, menu-driven 3D DESIGNER, with continuous, preprogrammed and coordinate-based drawing, with pan, scroll, translate, rotate about any axis (not just the x-y-z axes, and not just in awkwardly big stens!), recolour, rescale, recentre, zoom edit/delete (points, lines and 'positions'), element reposition, stretch, multispeed. Fully rubber-banded autoogenerics are provided. Defaults are redefinable via a configurator - this is very flexible, even the keys to be used for various manipulations may be reassigned!! Any shape can be coped with - no requirement for any regularity / unified variation of cross section. All modes, screen colours, ink colours, windows are supplied. Screens (ie. 2-D projectione either in normal mode and in XOR. Comprehensive file management utilities are supplied. Screens (ie. 2-D projectione either in normal mode and in XOR. Comprehensive file management utilities are supplied. Screens (ie. 2-D projectione and beaution of cross section. All modes, compatible) of file for subsequent editing or adjusting with (say) Ete-G. 2-D-D is and beaut.
The second program is an easy-to-use 3D TOULKIT, adding hosts of ultra high speed commands to SuperBASIC. Many examples are provided in the excellent manual. You too can now achieve the absolutely spectacular 3-D animation you see on TV, without being an expert programmer on having any knowledge at all of machine code. In repaidity of line-drawing is to be seen to be believed, the screen's having any knowledge at all of machine code. In repaidity of line-drawing is to be seen to be believed, the screen's naving any knowledge at all of machine code. In repaidity of line-drawing is to be formation of commands added by the tolkit to SuperBASIC formation of commands added by the tolkit to SuperBASIC formation of complex 3-D special effects can be achieved, as well as technical or artistic 3 llustration

The system provides facilities for artistic and technical drawing and modelling in 3-0, with excellent aids for conceptualisation. It also gives you the power to achieve all the effects and controls in your own BASIC, BIGITAL C or assembler programs. 3D PRECISION costs just £49.95 complete with a large and lucid A4 manual.

4.2.2 The Standard Jump Block

The jump block for BIOS is of standard form with three new jumps added to allow utilisation of QDOS devices and access to the 68000. Thus the jump block is organised as follows:

```
Initialisations:
          $00 - hard restart
          $01 - soft re-boot
I/O channels:
          $02 - console status
          $03 - console input
          $04 - console output
          $05 - printer output
          $06 - paper tape punch
          $07 - paper tape reader
Drive control:
          $08 - set read/write head to track 00
          $09 - select drive
          $0A - select track
          $0B - select sector
Disk I/O:
          $0C - set data buffer address
          $0D - read sector
          $0E - write sector
Other functions:
          $0F - inquire printer status
          $10 - translate record number
BIOS extensions(available in SUCCESS only):
          $11 - device allocation
          $12 - byte/string device I/O
          $13 - execute 68000 code in Z80 memory
```

4.2.3 BIOS Jump Block Extensions for SUCCESS

```
QDOS$ALLOC- BIOS call $11
```

This routine allows a single device to be allocated or machine resources to be set up. In all cases except SETBAUD, HL points to the filename and MUST BE EVEN.

\$14 - define function keys

The Accumulator contains a key, one of:-

On return the Accumulator contains either 0 (if successful) or the standard QDOS error keys.

Note: The channel is closed on a soft or hard system reboot.

4.2.4 Special Extension: DEFKEY

Another extension has been added but is not available from within BIOS and MUST therefore be called using an OUT instruction:

DEFKEY - OUT (\$14),A

This routine will allow a function key to be defined as a string of characters. The key to be defined is held in the accumulator. The key values are given in section 3.6.

The HL register pair should point to an area of memory which contains the definition (0 if the definition is to be deleted). The definition can be any valid characters finished by a NULL (code 0) character. The maximum length of the definition should be no more than 128 characters, any more being normally ignored.

4.2.5 Defining drive G:

When the DEFKEY routine is called with A=255, drive G is to be defined. HL points to an area of memory containing a definition (if HL=0, the definition is deleted). Drive G can only be used after it has been defined (else BDOS will return a "Bdos Err On G: Select"). To use this function, you will need to know a lot about the format of the disks to be read or written to.

WARNING! Special care is necessary if drive G: is to be used for raw-disk writing (as opposed to read-only operations). If not correctly set up in this mode, a QDOS disk may easily have its format destroyed. Do test it with a blank disk first - Digital Precision will accept no responsibility for any data corruption caused by use of this facility. Before using a disk in drive G:, press CTRL-C to ensure that it is correctly logged on. Similarly, if data on the disk has been altered, press CTRL-C BEFORE you remove it to ensure that the data is correctly written to it.

Note that the user also has to define the disk parameter block (the parameter block for drive G: is located at \$EADC in Z80 memory, using the standard CP/M environment). The disk parameter header is set for this drive and cannot be altered (nor is there any reason why it should).

The disk parameter header contains the following data:

SPT (2 bytes) - Logical sectors per track (each sector is 128 bytes long)

BSH (1 byte) - Block Shift Factor, ie; that power of two which gives the number of logical sectors in each block on disk (2 raised to the power BSH = number of logical sectors per block)

BLM (1 byte) - Block allocation mask = BSH - 1

EXM (1 byte) - Extent mask, ie. the number of extents per directory entry

DSM (2 bytes) - Number of blocks that are physically present on disk DRM (2 bytes) - Number of directory entries per disk

ALO (1 byte) and ALl (1 byte) - Initial allocation table contents (so that directory information will not be overwritten)

CKS (2 bytes) - Number of checked directory entries

OFF (2 bytes) - Sector offset, ie. number of tracks to be missed initially

```
(Note: Comments are preceded by an '!', and there must be no padding
spaces at the start of a line.)
<number> ! SPT, logical sectors per track
<number> ! BLS, block size: = 1 for 1024-byte blocks,
                            = 2 for 2048-byte blocks,
                            = 3 for 4096-byte blocks,
                            = 4 for 8192-byte blocks,
                            = 5 for 16384-byte blocks.
<number> ! DSM, directory storage mask: -1
<number> ! ALV, allocation vector: address of a scratch pad area to be
          used by BDOS to keep disk storage allocation information.
<number> ! OFF, track offset
! Now the data needed by raw-disk read/write routines
<number>
                 ! DRV, drive to be raw-read/written to
                 ! PSS, physical sector size
<number>
                 ! NPS, number of physical sectors
<number>
<number>
                 ! NPT, number of physical tracks (usually 40 or 80)
<number>
                 ! DSF, disk stepping factor
<number>
                ! FLG, see above for values
! Now follow with a list of the order of sectors on the disk
<number>, NPS numbers
```

The file contains the following data:-

Look at the *.DRV files on the master disk for examples of what they should look like.

4.3 CP/M-Z80 Disk Structure

BIOS utilises a large QDOS-format file called CPMFILES, which contains all of the CP/M files except those on an active drive G:. This permits many separate devices to contain the CP/M programs.

The CPMFILES file contains each 128-byte logical sector in logical order. Each logical track holds 20 logical sectors, with a varying number of logical tracks (210 on a 512K drive).

Drive capacity could have been increased, but then most freely available CP/M disks could not be easily converted to run on the system.

Ε																3.1.3.5
												-		•	-	•
	•										•	. •	•	•		
														-		•
														-		
END LINE								-	-	•		٠		•	. •	. 3.3.1
ENTER KEY			_													. 3.3.1
	· ·															
EXTENSION		•		•	•	•	•	•	•	•	•	•	•	•	•	. 3.1
																•
F			-			٠		•	•	-	-	-				3.1,3.5
FILENAMES			_													. 3.1
																3.1,3.6
FUNCTION KEYS																
FUNCTION REYS .		•	-	•	•	•	•	•	•	-	•	-	•	•	٠	. 3.0
G		-		•		+	•	-		-	-	-	•	•	•	3.1,3.5
G:, G>											2	,3.	1,3	.6,	4.2	.1,4.2.5
GO68000	_	_	_		_					_	_					. 4.2.3
		•	•	•	•	•	•	•	•	-	•	•	•	•	٠	
**																2 -
н													-		•	
																. 3.1
HORIZONTAL TAB .											•					. 3.3.2
I																. 3.5
	• •	-	_	-							-	•	•			
INSERT												•	•	•	-	
INTERPRETER .		-	•	•	٠	•	•	-	٠	•	-	•	٠	-	•	. 4.1
JUMP BLOCK .						_									4.2	.2,4.2.3
к					_						_					. 3.5
7	-				_							•				
KEY			•	٠			•			-		•	•			3.1,3.6
KEYPAD.KEY .		•	•	•	•	•	•		•	•	-	-	•	•	•	. 3.6
_																2.5
L					•				•		•	•	•	•	•	
LIB										•			-	•		. 3.1
LINE FEED .				_			_									. 3.3.1
LINE NUMBERING																. 3.5
****	-															
LOAD		•	•	-	•	•	•	•	٠	•	•	•	•	•	•	. 3.5
м			•	•	-	-	-	•	-	-	•	•	٠	•	•	. 3.5
MAC															-	. 3.1
MACHINE CODE.			_	_	_					_	_	_	_	_		4
 	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	_ -
NT IN BATTO																2 1
NAME		-	-	•	•	•	•	•	•	•	٠	•	•	•	-	. 3.1
NEW LINE		-	-	-	•	•	•	•	•	•	•	•	-	•	-	. 3.3.1
0		-	-													. 3.5
OUT INSTRUCTION				_		_		_							4.1	.1.4.2.3
ov																. 3.5
· · · · ·		•	•	•	•	•	-	•	•	•	•	•	•	•	•	
D1D1W0000 0:																4 2 5
PARAMETER BLOCK	•	-	•	•	•	•	•	•	٠	•	•	-	•	-	-	. 4.2.5
PAUSE		-	-	-	-	•				-		•	•	-	•	. 3.5
PERIPHERAL DEVI	CES	_	-	-											3	.2,4.2.1
PIP					_											. 3.5
PIRATING	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	. 1.2
·		-	-	-	•	•	•	•	-	•	•	•	•	•	, ,	
PRINTER		-	-	+	•	-	•	•	•	٠	•	-	-	5.	1,3	.2,3.3.2
PRN		-	-	-	+	-	•	-	-	•	•	•	-	-	•	. 3.1
PROGRAMS			-			-										. 1.3
PTP					_		_									. 3.2
	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	- *
0																. 3.5
Q	• •	•	•	•	-	•	•	-	•	•	•	-	•	•	•	
QDOS\$ALLOC .		-	-	-	•	•	٠	•	-	+	•	+	-	•	•	. 4.2.3
QDOS\$UTIL .			•	•	•		•	٠	•			•	-		•	. 4.2.3

R	_		-		-		-		-	•	-	•		•	•					3.5
RAW-DISK					•		•		•	-		•								,4.2.5
	-		•	•				•	•	-	•	•	-	-					•	3.6
RDR .	•	•				•	•	•		•	•	•	•	-	•	•	•	•	٠	3.2
REFERENCE	Е ВО	OKS		-	-				-			•		•				•		1.3
REGISTER	S		•																	4.2.3
REN .	_																			3.4
RETURN																		_	_	3.3.1
RETYPE	•		:										Ċ		•					3.3.1
RETIFE	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	
s																				3.5
	•									٠	•		-				٠		٠	
SAVE .			•										-		_	-		•		
SCREEN P									•				•		-		٠			3.5
SCROLL L										•		-	-	_	•		٠	_		3.3.2
						•	•		-		-	•	•	•	•	•	•	•	٠	3.6
SINGLE S	TEPP	ING	;			-							-							3.5
SOFT REB	TOC				_					٠										3.3.1
SOFTWARE						_			_						_			-	_	1.3
STAT .													_							.3.5
SUB .																				3.5
SUBMIT				-	-	-	_	_	_	_	_	_	_	_	_					3.5
																				3.1
																				3.4
SYSTEM C	OMMA	מטמ.	1	•	•	٠	•	•	•	•	•	•	-	-	•	•	•	٠	٠	3.4
T																				
TAB .																				
TEMPORAR'	Y FI	LE	-	-	-	-	-		-							•	-	-	-	3.5
TERMINAL	EMU	LAT	'ION	I				-	_						-			-		3.7
TEXT BUF	FER	-											4							3.5
TEXT MOD																				3.5
TRANSIEN'																				3.5
																				3.1
																				3.4
TIPE .	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	2.9
																				3.5
U	_	•	-	-	-	-	-	•	•	•	+	•	•	•	-		-	•		
UNDERLIN	_	-	•	-	-	•	•	•	•	•	•	•	•	-				•		
UNDO .	•	-		•	-	•	•		•	•	•	•	•	-	•					3.3.1
UPPER CA	SE	•	•	•					-			•				•		•	•	2,3.5
USER .												-			-		•			3.4
v			_		_	_	-				-									3.5
VT52 .		_		_					_								_	_		3.7
w																	_		_	3.5
WRQDOS	•	•	•	•	•	•	•	•	•	•	•	٠	•	•	•	-	•	•	-	3.6
Migboo	•	-	•	-	-	•	•	-	-	•	•	•	•	•	•	•	-	-	-	
х																				3.5
	•	•	-	•	-	-	•	-	•	•	-	-	•	•	•	•	٠	•	٠	
XSUB .	-	•	•	-	•	•	•	-	•	-	-	-	٠	•	•	•	•	•	•	3.5
_																				2.5
z	•	•	•	•	٠	•	•	•	•	•	•	•	•	•	٠	•	•	•	•	3.5
																			_	4 5 5
68000 PR	OCES	SOR	Ł	•	-	•	•	-	-	-	•	•	•	•	•	4	ι,	4.2	. 1,	4.2.3
\$\$\$.	-	-	•	•	•	•	-	•	•	•	•	•	-	٠	٠	•	٠	-	•	3.1
*	•	-	•	•		•	-	-	-	•		•		•			•		-	3.1
?	-	-			•		-	-	_	-				•	•			-	٠	3.1
^								_										3	. 3.	1,3.5

7

A		٠				+			-	-	-	-	-				3.5	5
A:, A> .			. ,														2,3.6	5
ADDRESS .												-						
AMSTRAD DISK																	4.2.5	5
ANALYSE .																	2,3.6	
																	1,3.5	
ASSEMBLE .	• •																	
																	3.5	
ASSEMBLER FI	LES	•	•	• •	•	•	•	•	•	•.	•	•	•	•	1	, ک	3.1,4	ł
в	• •	-												•	•	-	3.5	
B:										٠		•		•		•	3.6	
BACKSPACE KE	Υ.											•					3.3.1	
BACKSTEP .					-				•				•				3.3.1	L
BACKUP .									-		-		_		_	-	1.2	2
																3.	1,3.5	5
BAUD RATE																_	.3.6	5
BBC DISKS																	4.2.5	
																	1,4.2	
DIOD		•		•	•	٠	•	•	•	•	•	-	•	•	•	4,	1,4.2	۵
с																2	1 2 5	-
																	-	
							-										3.3.1	
CCP							•											_
CLONE_BAS																	1.2	
COM		•			-	•			-	-	-	-			•	-		
COMMANDS .																	3,3.5	
																	3.2	2
CONTROL CODE																		
COPYING .															1.2	,3.	4,3.5	5
CP/M																	2,4.1	
CPM CONFIG B															3	.1.	4.2.1	L
CPMFILES .																		
***************************************		•		•	•	•	•	•	•	•	•	•	•	•	•	•	0,	
D																3	1 2 5	
							•						•					
							:					•				_	3.6	
DDTZ	• •	•		•	•	•	-	-	-	•	•	•	•	•	•	•	٠. ٠	
DEBUGGING	• •	•		•	-	-	•	•	•	•	•	•	•	•	•	•	.3.5	
DEFINE .		٠		•	•	•	•	•	•	•	-	•	•	-	2,3	6,	4.2.5	
DEFAULTS.KEY	•	•			•	•			•	-	•	-	-	-	•	•		
DEFKEY .		-		•	-	-					-		-		3.	6,	4.2.3	\$
DELETION OF	CHARA	CTER	s.		-		-			_		_			_	-	3.5	į
DELETION OF	FILES										_	-				-	3.4	ļ
DELQDOS .										_		_					3.6	,
DEVICES .															3.	.2.	4.2.1	
DIR		_									_						3.4	
DIRECTORY		-	- ,	•	•	•	•	•	•	•	•	:	•	•	•	_	4,3.6	
DISK-BASED S	YSTEM			•	•	•	٠	•	-	•	•	٠	•	٠	•		1.1	
DISK STRUCTU		•	• •	•	•	•	•	•	•	•	•	•	•	•	•	•	4.3	
DOC	13E -	•		•	•	•	•	•	•	•	•	•	•	٠	•	•		
		•		•	•	•	•	•	•	-	•	•	•	•	•	•	3.1	
DRIVES .	- +	•		•	•	•	•	-	•	•	-	•	-	-	-		3.1	
DRV		•	•	•	•	•	•	•	-	-	•	•	-	•	•		1,3.6	
DUMP		•		•	•	•	-	•	•	•	•	-	-	•	•		3.5	

When the routine to define drive G is called, HL points to a raw-disk parameter block. This contains the following data:

RAW\$DRV (1 byte) - Drive number to be used

= 1 for FLP1, 2 for FLP2, 3 for FLP3, 4 for FLP4

RAWSPSS (1 byte) - Physical sector size

= 0 for 128, 1 for 256, 2 for 512, 4 for 1024 bytes.

RAWSNPS (1 byte) - Number of physical sectors

RAWSNPT (1 byte) - Number of physical tracks

RAWSDSF (1 byte) - Number of steps in between tracks-1

This is to allow 40-track disks to be read/written to in an 80-track drive (in that case it would be 1, normally 0)

RAW\$FLG (1 byte) -

If bit 0 = 1, this is a double-density disk

If bit 1 = 1, this is a double-sided disk (not used)

If bit 2 = 1, then side 2's track numbers are reversed (this is the case with BBC disks)

RAWSWKSPC (2 bytes) - Pointer to a work space, which should

be the same size as a single physical sector

RAW\$LPSMAP (NPS bytes) - logical-to-physical sector map - included because some disks have the sectors interleaved to make disk disk operation faster, but required even if there is 1:1 sector interleaving

Since this is an exceptionally useful feature (for extracting programs from other CP/M disks), a utility program has been included to help configure drive G:. This utility (called DEFINE, see section 4.2) requires a file containing parameters, viz. the absolutely essential details needed for most of the CP/M formats (any weird formats would require a machine-code defining program). Some of the more common definitions are supplied as files: the BBC CP/M-format disks (double-sided, 80-track), Amstrad CP/M-format disks (both data-and system-format disks, single-sided, 40 tracks; this assumes that a 40-track 3" drive is plugged in).

This routine allows data to be transferred to and from a device. On entry the Accumulator should contain:-

- 0 IOSFBYTE Read byte from device (returned in register C)
- 1 IO\$FSTRG Read string of characters:

HL = Base address

BC = Length (amount read on return)

- 2 IO\$SBYTE Write byte to device (in register C)
- 3 IO\$SSTRG Write string of bytes:

HL = Base address

BC = Length (amount written on return)

On return the A register contains either 0 (if successful) or the standard QDOS error key.

G068000 - BIOS call \$13

This routine will execute a piece of 68000 code in Z80 memory. The user must ensure that 68000 registers A2 to A6 and D3 to D7 are not modified.

HL - Base of program

BC - \$4AFB (identification key)

If the HL register has an odd value or the BC register does not contain the identification key, a message will be given and the system will automatically perform a soft reboot.

The 68000 registers have the following values:

D3.W - DE registers (read only)

D4.W - BC registers

D5.W - HL registers

D6.B - F register D7.B - A register

- Channel ID of main display

Al - Pointer to Z80 execution loop

A2 - Pointer to internal data

- Z80 stack address A3

- Base of workspace A4

- Base of Z80 memory A5

- Next instruction to execute

To get the value of the Z80 stack pointer, subtract A5 from A3. Since any modification of registers Al-A6 would be fatal, they are preserved before the routine is called and restored afterwards.

All of the BIOS extensions need not be called using the jump block. Like all BIOS routines, they can be called using the Z80 OUT instruction.

For example: OUT (\$11), A will execute the QDOS\$ALLOC routine.

4.1 How SUCCESS Is Organised

The emulator consists of two parts:

- the Z80 interpreter, which does all the hard work;
- the CP/M operating system proper, which comes in three sections:
 - BIOS (Basic Input/Output System), i.e. the hardware-dependent piece of code, which is invoked when a Z80 IN or OUT instruction is used and does all the communication between the Z80 (pseudo) and 68000 processors. It is this part that would have to be altered if any other system but CP/M was to be emulated
 - BDOS (Basic Disk Operating System), which utilises BIOS to provide file handling
 - CCP (Console Command Processor), the part the user communicates with

When the Emulator is loaded, it will search for a file called BIOS_CDE on drive FLP1. If this (68000 code) file is not found, an error is given and Z80 interpretation is aborted.

Once BIOS is loaded and executing, it will attempt to open a file called FLP1_CPMFILES. If unsuccessful, it will wait indefinitely, requiring the QL to be reset. On successfully opening the file, BDOS and CCP are loaded into memory. This done, CP/M will be executed.

The CCP - This is what the user communicates with

4.2 BIOS

4.2.1 Resources

BIOS for this system consists of a series of Z80 OUT instructions and some data needed within the Z80 environment. The OUT instructions are used to "hook" out to the 68000 processor. When an OUT is executed, a piece of 68000 code (held in the file BIOS_CDE on disk) takes over for maximum speed and easy interfacing to the QDOS operating system.

BIOS provides the user with the seven disk drives described in section 3.2.

Before Drive G can be used, it has to be configured. This may be done at machine-code level through one of the BIOS extensions as shown later.

For the assignment of the devices that CP/M allows to specific QDOS devices see section 3.2. The LST device can be redefined using the cpmconfig_bas program.

15

DEFKEY <keynum>,'string' DEFKEY <keynum> DEFKEY <keyfile>

Defines a function key. If there is no string, the definition will be deleted. Control characters within the string are defined as \chr> (<chr> standing for any upper-case character). If a filename is specified, an attempt is made to load this file, the default extension being '.KEY'. The file holds a series of lines containing a key number and a string definition each. It can be created using ED. The key numbers are:

Кеу	Normal	Ctrl	Shift	Ctrl+Shift
Fl	232	233	234	235
F2	236	237	238	239
F3	240	241	242	243
F4	244	245	246	247
F5	248	N/A	250	251

(On the Thor F6 to F10 correspond to Shift + F1 to F5 respectively.)

Arrows U	P	Down	Left	Right
224	225	226	227	

e.g.: DEFKEY 232, 'DIR A:\M'
DEFKEY 236, ''
DEFKEY DBASE.KEY

A number of key assignment files are already provided, viz.

DBASE.KEY DEFAULTS.KEY KEYPAD.KEY

DELODOS <qdosname>

Deletes a QDOS file after asking for confirmation, e.g.
DELQDOS FLP1 CPMSYSTEM

FORMAT

Asks for the drive to be formatted and then creates the CPMFILES file on it, provided the medium has been previously been formatted at the SuperBASIC level (this is for safety reasons). Also performs system regeneration on drive A: and B:.

RDQDOS <cpmname> <qdosname>

Copies a QDOS-format file to a CP/M format file. If the CP/M file exists or the QDOS file does not exist, the copy will be aborted.

e.g.: RDQDOS INTRO.DOC FLP1_INTRO_LIS

λl

STAT <command line>

Supplies information about drives. The STAT program recognises the following sub-commands:

STAT

Supplies information on write-protection of drives and free space thereon

STAT <drive>:

Supplies information on free space on the disk in <drive>

STAT <filename>:

STAT <wildcard>:

Supplies information on size of files, file characteristics and free space on disk

STAT <filename>\$<characteristic>

STAT <wildcard>\$<characteristic>

Sets or unsets file characteristics

<characteristic> may be:

DIR - make into non-system file

R/O - read only

R/W - read/write

SYS - make into system file

STAT <device>:=<defined device>:

Assigns a pre-defined device name to one of the four device names (BAT/CRT/TTY/UCl to CON, CRT/LPT/TTY/ULl to LST, PTP/TTY/UPl/UP2 to PUN, PTR/TTY/URl/UR2 to RDR)

STAT <drive>:=R/O

Write-protects <drive>

STAT DEV:

Lists device assignments

STAT DSK:

STAT <drive>:DSK

Tabulates properties of currently active disks

STAT USR: Tabulates user areas on disk and indicates current

user number

STAT VAL: Tabulates valid STAT sub-commands and arguments

SUBMIT <filename>

SUBMIT <filename> <argument list>

Loads and executes a file consisting of CP/M commands. The command file may include variables of the form \$1, \$2 etc, which will be replaced with the elements of the (space-delimited) argument list. The command file may contain comment lines starting with any of the characters <>.,;:=?*_ or with either a left or right curly bracket.

SYSGEN

does not work; nor is it needed, the operating system being already on the working disk

XSUB

(in command files only:)

Loads and executes one line from the temporary file created by SUBMIT whenever the running program expects console input

H<value1>,<value2> ('hex compute') Computes both <value1>+<value2> and <valuel>=<value2> in hex (up to 4 digits) I<filename> ('input') Selects <filename> for next input operation L<start>,<end> ('list') Lists command mnemonics between <start> and <end> addresses M<start>, <end>, <destination> ('move') Copies memory block between <start> and <stop> addresses to destination R<offset> ('read') Reads a file whose start address is <offset> bytes from the current program address ('set') S<address> Shows (and permits modification of) byte value at <address> in hex T<number> ('trace') Executes <number> program steps, showing processor status after each step U<number> ('untrace') Ditto, but shows processor status after last step Х ('examine') Shows processor status (flags, registers, current command) X<register> Shows (and permits modification of) the value of <register> DUMP Does not work; use DDTZ ED <filename> Edits a file and produces a backup of its last version under the name <drive>:<name>.BAK. The following sub-ommands are recognised within the ED program: Adds lines of text from original file to text nΑ buffer Writes lines of text from the text buffer into the n₩ temporary file Writes sections of text into a temporary file nΧ Inserts text from the temporary file at the text R pointer position R <filename> Inserts text from file at the text pointer position Moves text pointer to start or end of text +/-B Shifts text pointer by n characters +/-nC Shifts text pointer by n lines nF<text> Finds n-th occurrence of string nN<text> Ditto, but reads new lines from original file as required Shifts text pointer and types line of text pointer +/-n +/-nTTypes n lines of text Types n screen pages and shifts text pointer 1 Switches to text mode

I<text>^Z Inserts <text> in command mode

I<text><Enter> Ditto, including end-of-line marker

3.2 Peripheral Devices

The devices that CP/M allows have been assigned to specific QDOS' devices as follows:

Device CON: As CON 480X250A16X3

Device LST: As SERIHR - changeable by means of the cpm_config

program

Device PTP: User-definable, device asked for Device RDR: User-definable, device asked for

These names may occur in various CP/M commands.

3.3 Control Codes

The following control codes are recognised. The ^ sign stands for combination with the CTRL key; e.g., ^X means that CTRL and X must be pressed simultaneously.

3.3.1 Used on input:

^C (first on line:) terminates command execution, causes soft re-boot. Use this if '>' prompt does not appear after an error message

Note: Use CTRL + SHIFT + C to change between processes (like CTRL + C (Thor: Sys Req) under QDOS.

- ^E ('end line':) starts a new physical line on screen; no change is made to the logical line
- 'H ('backstep', also backspace key): deletes last character both from the screen and from the input buffer
- ^J ('line feed') and
- ^ M ('return'; also Enter key): terminates an input line and passes input buffer contents to program
- ^ R ('retype':) appends a hash sign to current input line, abandons this line and retypes current contents of input buffer on next line
- O ('undo:) appends a hash sign to current input line, abandons this line, empties the input buffer and starts a new input line
- ^ X ('cancel':) removes current input line from the screen and empties the input buffer

3.3.2 Used on output:

- ^I ('horizontal tab'; also Tab key): generates the number of spaces required to take the cursor to the next column number divisible by 8
- ('printer':) toggles printer (device LST) echoing of screen output. Printer echoing will be switched off by a soft re-boot.
- AS ('scroll lock':) toggles screen scrolling

RUNNING CP/M SOFTWARE 2.

- Obtain the program from a CP/M library, or buy it. Don't pirate.
- Transfer the disk's contents to QDOS:
 - Place your working disk of SUCCESS in drive 1 and the program disk in drive 2
 - Boot up the system, or enter LRUN flpl boot
 - You will be asked if it is running on a Thor-20 or a QL/Thor (this is because of some slight incompatibility between the 68000/8 and the 68010/20/30)
 - When the A> prompt appears, enter the following:
 - If the format of the library disk is known to be one of the following, enter the appropriate command from the following list:
 - DEFINE AMS-DATA.DRV (if the disk in drive 2 is in Amstrad data format) or
 - DEFINE AMS-SYST.DRV (if it is in Amstrad system format) or
 - (if it is a BBC micro Z80 second-- DEFINE BBCZ80.DRV processor disk)
 - If the format of the library disk is not known, obtain it as follows:
 - Enter ANALYSE and follow subsequent prompts
 - When the A> prompt reappears, enter RDODOS DISKFMT.DRV FLP1 DISKFMT_DRV
 - On completion of this command, enter DEFINE DISKFMT

CP/M is case-blind; you may prefer to use lower case.

- Check the disk format by loading and running one of the programs on the disk, making sure that it performs NO WRITING to disk; if the format turned out to be incorrect, writing would corrupt the disk. If the format does not seem to be correct, go back to the ANALYSE command and alter some of the parameters it asks for.

Once the disk format is established as correct, it will be possible to access all files on the program disk under the virtual drive name G:

- Type G:
- When the G> prompt appears, follow the instructions for the program in question. In many cases this will involve no more than typing in the program name.

Terminal Emulation 2.3

Most CP/M programs are designed to be used on a system which emulates a terminal, most commonly a VT52. SUCCESS supports most VT52 emulation sequences, viz:

- [ESC]A Moves cursor up one line
- [ESC]B Moves cursor down one line
- [ESC]C Moves cursor right one character
- [ESC]D Moves cursor left one character
- [ESC]E Clears entire screen
- [ESC]H Homes cursor
- [ESC]J Erases to end of screen, including current line
- [ESC]K Erases to end of line
- [ESC]Yrc Moves cursor to (r-32), (c-32)
- [ESC]p Inverse video
- [ESC]q Normal video

1.3 Copyright Notice

SUCCESS, like the rest of DP's programs, is not protected against unauthorised copying and theft. This has obvious advantages for you, in that you can make backups freely and transfer the program suite freely from one medium to another. But the absence of copy protection has a big disadvantage for us: it means that every time you sell or give away a copy of SUCCESS to someone, we lose a sale that would have helped us to fund more development of QL software.

The only part of the package that may be copied freely is the CP/M operating system (version 2.2), which is in the public domain. No cash advantage should be asked for or accepted on its redistribution.

There is no QL software publisher so large and committed as DIGITAL PRECISION. But the flow of new QL software from DP, and the availability of older titles, depends upon the honesty of a circle of buyers.

DIGITAL PRECISION will remain active in the QL market for the foreseeable future, but future products can only appear if people are willing to pay for them.

In effect, against the trend - especially on top-quality products - we have taken a gamble on your honesty. Please don't let us down.

We offer rewards for information which enables action (civil and criminal) to be taken against software pirates, big and small. Of course, we hope to spend our time more productively!

1.3 How to Succeed with SUCCESS

There are several distinct uses you may want to make of SUCCESS. Sorted by level of sophistication, they are:

- (a) running any CP/M program in your possession on the QL or Thor;
- (b) using CP/M commands, mainly for some things that cannot be done as easily with QDOS and SuperBASIC;
- (c) writing and running your own Z80 assembler programs.

Operating instructions for (a) will be found in Chapter 2; for (b), in Chapter 3; and for (c), in Chapter 4. However, this manual cannot be a substitute for a full treatment of the CP/M operating system and its handling. Getting such a book is suggested for level (a), recommended for level (b), and imperative for level (c). Here are two titles you may find useful:

B. Brigham CP/M Programmer's Encyclopaedia Que

Peter Gosling Using CP/M Macmillan

SUCCESS for the Sinclair QL

CONTENTS

L.	INTRODUCTION
1.1	Why an Emulator?
1.2	· Getting Started
1.3	How to Succeed with SUCCESS
2.	RUNNING CP/M SOFTWARE
3.	SUCCESS AT THE CP/M COMMAND LEVEL
3.1	Filenames
3.2	Peripheral Devices
3.3	Control Codes
3.3.1	Used on input
3.3.2	Used on output
3.4	Operating System Commands
3.5	Standard .COM-File (Transient) Commands
3.6	Special SUCCESS Commands
3.7	Terminal Emulation
4.	SUCCESS AT THE MACHINE-CODE LEVEL
4.1	How SUCCESS is Organised
4.2	BIOS
4.2.1	Resources
4.2.2	The Standard Jump Block
4.2.3	BIOS Jump Block Extensions for SUCCESS
4.2.4	Special Extension: DEFKEY
4.2.5	Defining Drive G:
4.3	CP/M-Z80 Disk Structure

R			-		•	-	-		-	•									.5
RAW-DISK READ	/WRIT	ſΕ	•					-	-			_			3.5	,3.	6,	4.2	.5
RDQDOS										_								3	. 6
RDR																		3	. 2
REFERENCE BOO	KS																		. 3
REGISTERS .			:		:	:	:		:		:							4.2	
		•				_				_	_						•		.4
REN	-	•	•	•	•	•	•	•	•	•	-	•	•	•	•	•	•		
	•	•	•	•	•	-	•	+	•	•	•	•	•	•	•	. •		3.3	
RETYPE	•	•	•	٠	•	•	•	•	•	•	•	•	•	•	•	•		3.3	. 1
s											٠								.5
SAVE																	-	3	. 4
SCREEN PAGES																		3	.5
SCROLL LOCK .																	. :	3.3	. 2
SETBAUD		-																3	.6
SINGLE STEPPI																			.5
		-	_																
SOFT REBOOT .		•	•	•		٠	•	•	•	•	•	•	•						.3
SOFTWARE			-	•	•	•	•	•	•	-	•	•	-	٠					
	*			•		•	-		٠	٠	•	•	•	•	•		-		.5
SUB		•	•	•	•	•	•	•	•	•	•	•	-	•	-	•	•		. 5
SUBMIT				•						-	•	•	-	-	•	•	•		. 5
SYSGEN														-	_		_	3	.1
SYSTEM COMMAN	IDS																	3	. 4
т																		3	.5
TAB	-	_	_						:			:						3.3	
									-					•			•		.5
TEMPORARY FIL									•	•	-	-	-	•	•	•	•		
TERMINAL EMUL									•		•	-	•	•	•		٠		. 7
TEXT BUFFER .			•					•	•	•	•	•	•	•	•				. 5
TEXT MODE .										•									. 5
TRANSIENT COM	MAND				_	_	_							_				3	.5
TXT																		3	. 1
TYPE					_						_							3	. 4
	•	-	-	Ť		•	•	•	•	•	_	_					-		
U																		3	.5
	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•		•		.1
UNDERLINE .	•	•	•	•	٠	-	-	-	•	•	•	•	•	•	•	•	•		
UNDO	•	٠	•	•	٠	•	•	•	•	•	•	+	•	•	•	•		3.3	
UPPER CASE .	•	•		•	•	•	•	•	•	٠	٠	•	•	•		•	•		
USER	•	-	-	•	-		•	•	•	•	•	•	•	•	•	•	•	3	. 4
v															-		-	3	. 5
VT52							_		_						_			3	. 7
w																		3	.5
WRQDOS	•	-	-	•	-	•	-	-	•	-	•	•	•	•	•	•	•		.6
MKQDOS	•	•	•	•	•	•	•	•	•	•	•	-	-	•	•	•	•		
17																		2	.5
х	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•		
XSUB	•	•	•	•	•	•	•	•	•	•	•	•	•	•	+	٠	•	3	.5
Z												•	-				-	3	. 5
68000 PROCESS	OR													4.	1,4	.2.	1,4	1.2	. 3
\$\$\$	_																		.1
*	-	-	-	-	-	-	-	-	-	-	-	_	-		_		-		.1
	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	-	•		.1
	•	•	•	•	٠	•	•	•	•	•	•	•	•	•	•	• ¬	3.1		
**	_	_	_	-	_	_	_		_	_	_					٠.	J	الاما	. ၁

Α							-				-	-									3	3.5
A:, A>																					2,3	3.6
ADDRESS																						3.5
AMSTRAD I	oisi	(S																				2.5
ANALYSE						•							•									3.6
ASM .		Ī						:		:			•									3.5
ASSEMBLE	•	•	•			•							•									3.5
ASSEMBLE		• [T TD C		•														•				
ASSENDED	X E3	LLES	>	-	•	•	•	•	•	•	•	•	•	•	-	•	•		Ι.	٠,	J.1	.,4
_																						
В	-	•	•	•						•			-				•	. •		-		3.5
B:	•	•	•			•		•		•			•		•	•	•			•		3.6
BACKSPACE	E KE	EY	•	•		-				•			•		-						3.3	
BACKSTEP	٠	•	-		-	4							•		•					•		
BACKUP	-	•	-			•				•			•		•	-	•			•		2
BAK .			•			•						•								3.	1,3	3.5
BAUD RATE	2			-	+		-														. 3	3.6
BBC DISKS	3		_	•		-													3.	6,	4.2	2.5
BDOS .																						1.1
BIOS .																						1.2
						•	•	•	•	•	•	•	-	-	-	-	-	_			-,	
с																				3.	1 . 3	1.5
CANCEL.	•		•																			
CCP .	•																					1.1
CLONE BAS																						2
_	9					-														•		3.1
COM .	•																			•		
COMMANDS	•	•				•																
CON .	•	•																				
CONTROL (3.3
CP/M .						-																
CPM_CONFI	[G_E	BAS	•	•															3.	1,	4.2	1.5
CPMFILES								-	-				-		-	-	_			3.0	6,4	1.1
D								-				-								3	1,3	.5
DAT .									_		_									•	3	.1
DBASE.KEY	?								-												3	.6
DDT .										_										_	3	.5
DDTZ .	_			_								-				Ī	-			_		.5
DEBUGGING						Ī	_	-	-	-	_	_	•	•	•	•	•	•		•		.5
DEFINE	•	-	•	-	-	-	-	-	-	•	•	•	٠	•	•	•	٠	,	3 1	- 6 4		.5
DEFAULTS.	KEV	,	•	•	•	•	•	•	-	-	•	٠	•	-	•	•	•	2,				.6
DEFKEY	1/1	•	•	•	•	•	•	•	-	-	•	-	•	-	-	-	•	•				2.3
	·	*		•		•	•	•	•	•	•	•	•	•	•	•	•		J - 1	0,1		
DELETION				TEI	K.D	•	-	•	٠	•	•	•	•	-	7	•	•	•		-		.5
DELETION	QF.	F.T.	ıES	•	•	-	•	•	•	•	•	•	•	٠	•	٠	•	-		•		.4
DELQDOS	-	٠	•	-	-	•	•	٠	•	•	•	•	•	•	•	•	•	•	_	•		-6
DEVICES	•	•	•	•	•	•	-	•	•	•	•	•	•	-	•	•	-	,	3 . :	2,4		.1
DIR .	•	•		-	-		•	•	•	•	•	٠	•	•	•	٠	•	-		•		.4
DIRECTORY		-			•				-	-	-	-	-	٠	-	-	•			3.4	4,3	
DISK-BASE	ED S	YST	EM		•																1	.1
DISK STRU	CTU	IRE								٠		-		•					,		4	. 3
DOC .								_					_	*							3	.1
DRIVES	+	_																				.1
DRV .																	-					
DIV V							-	-	-				_			-		_	•	3.1	L.3	.6
DUMP .							•	•	•	•	•	•	-	•	•	•	•	•	-	3.1	۱,3 3	.6

When the routine to define drive G is called, HL points to a raw-disk parameter block. This contains the following data:

RAWSDRV (1 byte) - Drive number to be used

= 1 for FLP1, 2 for FLP2, 3 for FLP3, 4 for FLP4

RAW\$PSS (1 byte) - Physical sector size

= 0 for 128, 1 for 256, 2 for 512, 4 for 1024 bytes.

RAW\$NPS (1 byte) - Number of physical sectors

RAWSNPT (1 byte) - Number of physical tracks

RAWSDSF (1 byte) - Number of steps in between tracks-1

This is to allow 40-track disks to be read/written to in an 80-track drive (in that case it would be 1, normally 0)

RAW\$FLG (1 byte) -

If bit 0 = 1, this is a double-density disk

If bit 1 = 1, this is a double-sided disk (not used)

If bit 2 = 1, then side 2's track numbers are reversed (this is the case with BBC disks)

RAW\$WKSPC (2 bytes) - Pointer to a work space, which should

be the same size as a single physical sector

RAW\$LPSMAP (NPS bytes) - logical-to-physical sector map - included because some disks have the sectors interleaved to make disk disk operation faster, but required even if there is 1:1 sector interleaving

Since this is an exceptionally useful feature (for extracting programs from other CP/M disks), a utility program has been included to help configure drive G:. This utility (called DEFINE, see section 4.2) requires a file containing parameters, viz. the absolutely essential details needed for most of the CP/M formats (any weird formats would require a machine-code defining program). Some of the more common definitions are supplied as files: the BBC CP/M-format disks (double-sided, 80-track), Amstrad CP/M-format disks (both data-and system-format disks, single-sided, 40 tracks; this assumes that a 40-track 3" drive is plugged in).

This routine allows data to be transferred to and from a device. On entry the Accumulator should contain:-

- 0 IO\$FBYTE Read byte from device (returned in register C)
- 1 IO\$F\$TRG Read string of characters:

HL = Base address

BC = Length (amount read on return)

- 2 IO\$SBYTE Write byte to device (in register C)
- 3 IO\$SSTRG Write string of bytes:

HL = Base address

BC = Length (amount written on return)

On return the A register contains either O (if successful) or the standard QDOS error key.

G068000

- BIOS call \$13

This routine will execute a piece of 68000 code in Z80 memory. The user must ensure that 68000 registers A2 to A6 and D3 to D7 are not modified.

HL - Base of program

BC - \$4AFB (identification key)

If the HL register has an odd value or the BC register does not contain the identification key, a message will be given and the system will automatically perform a soft reboot.

The 68000 registers have the following values:

D3.W - DE registers (read only)

D4.W - BC registers

D5.W - HL registers

D6.B - F register D7.B - A register

- Channel ID of main display AO

A1 - Pointer to Z80 execution loop

A2 - Pointer to internal data

- Z80 stack address A3

- Base of workspace

- Base of Z80 memory

- Next instruction to execute

To get the value of the Z80 stack pointer, subtract A5 from A3. Since any modification of registers A1-A6 would be fatal, they are preserved before the routine is called and restored afterwards.

All of the BIOS extensions need not be called using the jump block. Like all BIOS routines, they can be called using the Z80 OUT instruction.

For example: OUT (\$11), A will execute the QDOS\$ALLOC routine.

4. SUCCESS AT THE MACHINE-CODE LEVEL

4.1 How SUCCESS Is Organised

The emulator consists of two parts:

- the Z80 interpreter, which does all the hard work;
- the CP/M operating system proper, which comes in three sections:
 - BIOS (Basic Input/Output System), i.e. the hardware-dependent piece of code, which is invoked when a Z80 IN or OUT instruction is used and does all the communication between the Z80 (pseudo) and 68000 processors. It is this part that would have to be altered if any other system but CP/M was to be emulated
 - BDOS (Basic Disk Operating System), which utilises BIOS to provide file handling
 - CCP (Console Command Processor), the part the user communicates with

When the Emulator is loaded, it will search for a file called BIOS_CDE on drive FLP1. If this (68000 code) file is not found, an error is given and Z80 interpretation is aborted.

Once BIOS is loaded and executing, it will attempt to open a file called FLP1_CPMFILES. If unsuccessful, it will wait indefinitely, requiring the QL to be reset. On successfully opening the file, BDOS and CCP are loaded into memory. This done, CP/M will be executed.

The CCP - This is what the user communicates with

4.2 BIOS

4.2.1 Resources

BIOS for this system consists of a series of Z80 OUT instructions and some data needed within the Z80 environment. The OUT instructions are used to "hook" out to the 68000 processor. When an OUT is executed, a piece of 68000 code (held in the file BIOS_CDE on disk) takes over for maximum speed and easy interfacing to the QDOS operating system.

BIOS provides the user with the seven disk drives described in section 3.2.

Before Drive G can be used, it has to be configured. This may be done at machine-code level through one of the BIOS extensions as shown later.

For the assignment of the devices that CP/M allows to specific QDOS devices see section 3.2. The LST device can be redefined using the cpmconfig_bas program.

15

DEFKEY <keynum>,'string' DEFKEY <keynum> DEFKEY <keyfile>

Defines a function key. If there is no string, the definition will be deleted. Control characters within the string are defined as \chr> (\chr> standing for any upper-case character). If a filename is specified, an attempt is made to load this file, the default extension being '.KEY'. The file holds a series of lines containing a key number and a string definition each. It can be created using ED. The key numbers are:

Key	Normal	Ctrl	Shift	Ctrl+Shift
F1	232	233	234	235
F2	236	237	238	239
F3	240	241	242	243
F4	244	245	246	247
F5	248	N/A	250	251

(On the Thor F6 to F10 correspond to Shift + F1 to F5 respectively.)

Arrows	Up	Down	Left	Right
224	225	226	227	

e.g.: DEFKEY 232, 'DIR A:\M'
DEFKEY 236,''
DEFKEY DBASE.KEY

A number of key assignment files are already provided, viz.

DBASE.KEY DEFAULTS.KEY KEYPAD.KEY

DELODOS <qdosname>

Deletes a QDOS file after asking for confirmation, e.g.
DELQDOS FLP1 CPMSYSTEM

FORMAT

Asks for the drive to be formatted and then creates the CPMFILES file on it, provided the medium has been previously been formatted at the SuperBASIC level (this is for safety reasons). Also performs system regeneration on drive A: and B:.

RDQDOS <cpmname> <qdosname>

Copies a QDOS-format file to a CP/M format file. If the CP/M file exists or the QDOS file does not exist, the copy will be aborted.

e.g.: RDQDOS INTRO.DOC FLP1_INTRO_LIS

N

STAT <command line>

Supplies information about drives. The STAT program recognises the following sub-commands:

STAT

Supplies information on write-protection of drives and free space thereon

STAT <drive>:

Supplies information on free space on the disk in <drive>

STAT <filename>:

STAT <wildcard>:

Supplies information on size of files, file characteristics and free space on disk

STAT <filename>\$<characteristic>

STAT <wildcard>\$<characteristic>

Sets or unsets file characteristics

<characteristic> may be:

DIR - make into non-system file

R/O - read only R/W - read/write

SYS - make into system file

STAT <device>:=<defined device>:

Assigns a pre-defined device name to one of the four device names (BAT/CRT/TTY/UC1 to CON, CRT/LPT/TTY/UL1 to LST, PTP/TTY/UP1/UP2 to PUN, PTR/TTY/UR1/UR2 to RDR)

STAT <drive>:=R/O

Write-protects <drive>

STAT DEV:

Lists device assignments

STAT DSK:

STAT <drive>:DSK

Tabulates properties of currently active disks

STAT USR: Tabulates user areas on disk and indicates current

user number

STAT VAL: Tabulates valid STAT sub-commands and arguments

SUBMIT <filename>

SUBMIT <filename> <argument list>

Loads and executes a file consisting of CP/M commands. The command file may include variables of the form \$1, \$2 etc, which will be replaced with the elements of the (space-delimited) argument list. The command file may contain comment lines starting with any of the characters <>.,;:=?*_ or with either a left or right curly bracket.

SYSGEN

does not work; nor is it needed, the operating system being already on the working disk

XSUB

(in command files only:)

Loads and executes one line from the temporary file created by SUBMIT whenever the running program expects console input

Computes both <value1>+<value2> and <valuel>-<value2> in hex (up to 4 digits) I<filename> ('input') Selects <filename> for next input operation ('list') L<start>,<end> Lists command mnemonics between <start> and <end> addresses M<start>,<end>,<destination> ('move') Copies memory block between <start> and <stop> addresses to destination R<offset> ('read') Reads a file whose start address is <offset> bytes from the current program address ('set') S<address> Shows (and permits modification of) byte value at <address> in hex T<number> ('trace') Executes <number> program steps, showing processor status after each step ('untrace') U<number> Ditto, but shows processor status after last step Х ('examine') Shows processor status (flags, registers, current command) X<register> Shows (and permits modification of) the value of <register> Does not work; use DDTZ DUMP ED <filename> Edits a file and produces a backup of its last version under the name <drive>:<name>.BAK. The following sub-ommands are recognised within the ED program: Adds lines of text from original file to text nA buffer Writes lines of text from the text buffer into the n₩ temporary file Writes sections of text into a temporary file nХ Inserts text from the temporary file at the text R pointer position R <filename> Inserts text from file at the text pointer position Moves text pointer to start or end of text +/-B Shifts text pointer by n characters +/-nC Shifts text pointer by n lines nF<text> Finds n-th occurrence of string nN<text> Ditto, but reads new lines from original file as required Shifts text pointer and types line of text pointer +/-n +/-nTTypes n lines of text Types n screen pages and shifts text pointer Ι Switches to text mode

I<text>^Z Inserts <text> in command mode

I<text><Enter> Ditto, including end-of-line marker

('hex compute')

H<value1>,<value2>

The devices that CP/M allows have been assigned to specific QDOS' devices as follows:

Device CON: As CON 480X250A16X3

Device LST: As SERIHR - changeable by means of the cpm_config

program

Device PTP: User-definable, device asked for Device RDR: User-definable, device asked for

These names may occur in various CP/M commands.

3.3 Control Codes

The following control codes are recognised. The ^ sign stands for combination with the CTRL key; e.g., ^X means that CTRL and X must be pressed simultaneously.

3.3.1 Used on input:

^C (first on line:) terminates command execution, causes soft re-boot. Use this if '>' prompt does not appear after an error message

Note: Use CTRL + SHIFT + C to change between processes (like CTRL + C (Thor: Sys Req) under QDOS.

- ^E ('end line':) starts a new physical line on screen; no change is made to the logical line
- ^H ('backstep', also backspace key): deletes last character both from the screen and from the input buffer
- ^J ('line feed') and
- ^ M ('return'; also Enter key): terminates an input line and passes input buffer contents to program
- ^ R ('retype':) appends a hash sign to current input line, abandons this line and retypes current contents of input buffer on next line
- ^ U ('undo:) appends a hash sign to current input line, abandons
 this line, empties the input buffer and starts a new input
 line
- ^ X ('cancel':) removes current input line from the screen and empties the input buffer

3.3.2 Used on output:

- ^I ('horizontal tab'; also Tab key): generates the number of spaces required to take the cursor to the next column number divisible by 8
- ^ P ('printer':) toggles printer (device LST) echoing of screen output. Printer echoing will be switched off by a soft re-boot.
- AS ('scroll lock':) toggles screen scrolling

RUNNING CP/M SOFTWARE 2.

- Obtain the program from a CP/M library, or buy it. Don't pirate.
- Transfer the disk's contents to QDOS:
 - Place your working disk of SUCCESS in drive 1 and the program disk in drive 2
 - Boot up the system, or enter LRUN flpl boot
 - You will be asked if it is running on a Thor-20 or a QL/Thor (this is because of some slight incompatibility between the 68000/8 and the 68010/20/30)
 - When the A> prompt appears, enter the following:
 - If the format of the library disk is known to be one of the following, enter the appropriate command from the following list:
 - DEFINE AMS-DATA.DRV (if the disk in drive 2 is in Amstrad data format) or
 - DEFINE AMS-SYST.DRV (if it is in Amstrad system format) or
 - (if it is a BBC micro Z80 second-- DEFINE BBCZ80.DRV processor disk)
 - If the format of the library disk is not known, obtain it as follows:
 - Enter ANALYSE and follow subsequent prompts
 - When the A> prompt reappears, enter RDQDOS DISKFMT.DRV FLP1_DISKFMT_DRV
 - On completion of this command, enter DEFINE DISKFMT

CP/M is case-blind; you may prefer to use lower case.

- Check the disk format by loading and running one of the programs on the disk, making sure that it performs NO WRITING to disk; if the format turned out to be incorrect, writing would corrupt the disk. If the format does not seem to be correct, go back to the ANALYSE command and alter some of the parameters it asks for.

Once the disk format is established as correct, it will be possible to access all files on the program disk under the virtual drive name G:

- Type G: 3
- When the G> prompt appears, follow the instructions for the program in question. In many cases this will involve no more than typing in the program name.

Terminal Emulation 2.3

Most CP/M programs are designed to be used on a system which emulates a terminal, most commonly a VT52. SUCCESS supports most VT52 emulation sequences, viz:

- [ESC]A Moves cursor up one line
- [ESC]B Moves cursor down one line
- [ESC]C Moves cursor right one character
- [ESC]D Moves cursor left one character
- [ESC]E Clears entire screen
- [ESC]H Homes cursor
- [ESC]J Erases to end of screen, including current line
- [ESC]K Erases to end of line
- [ESC]Yrc Moves cursor to (r-32), (c-32)
- [ESC]p Inverse video
- [ESC]q Normal video

1.3 Copyright Notice

SUCCESS, like the rest of DP's programs, is not protected against unauthorised copying and theft. This has obvious advantages for you, in that you can make backups freely and transfer the program suite freely from one medium to another. But the absence of copy protection has a big disadvantage for us: it means that every time you sell or give away a copy of SUCCESS to someone, we lose a sale that would have helped us to fund more development of QL software.

The only part of the package that may be copied freely is the CP/M operating system (version 2.2), which is in the public domain. No cash advantage should be asked for or accepted on its redistribution.

There is no QL software publisher so large and committed as DIGITAL PRECISION. But the flow of new QL software from DP, and the availability of older titles, depends upon the honesty of a circle of buyers.

DIGITAL PRECISION will remain active in the QL market for the foreseeable future, but future products can only appear if people are willing to pay for them.

In effect, against the trend - especially on top-quality products - we have taken a gamble on your honesty. Please don't let us down.

We offer rewards for information which enables action (civil and criminal) to be taken against software pirates, big and small. Of course, we hope to spend our time more productively!

1.3 How to Succeed with SUCCESS

There are several distinct uses you may want to make of SUCCESS. Sorted by level of sophistication, they are:

- (a) running any CP/M program in your possession on the QL or Thor;
- (b) using CP/M commands, mainly for some things that cannot be done as easily with QDOS and SuperBASIC;
- (c) writing and running your own Z80 assembler programs.

Operating instructions for (a) will be found in Chapter 2; for (b), in Chapter 3; and for (c), in Chapter 4. However, this manual cannot be a substitute for a full treatment of the CP/M operating system and its handling. Getting such a book is suggested for level (a), recommended for level (b), and imperative for level (c). Here are two titles you may find useful:

B. Brigham
CP/M Programmer's Encyclopaedia
Que

Peter Gosling Using CP/M Macmillan