SIZMON 3
documentation

Table of contents

History

v3.0 – 2011.12.31 – first public version
v3.1 – 2013.06.07 – assembly instruction set mnemonics modified to match the table at ffd2.org
v3.11 – 2014.02.06 – added info about number converter
v3.12 – 2014.07.28 – bugfix in E command that prevented entering editor again
v3.13 – 2014.07.29 – bugfix in $ command that prevented directory listing with JiffyDOS

Features

- ROM version: uses 16k ROM area and several bytes RAM for work variables and routines
- ROM configuration is autodetected (can be used in function low, c1lo and c2lo slots)
- Starts with pressing Escape during reset-routine
- Dynamic parameter input: every numeric parameter can be entered in hexadecimal (prefix: none or $), decimal (prefix +), binary (prefix: %), ASCII (PETSCII, prefix: ") or character screen code (prefix ') form. Only the least significant digits are used (8 or 16 bit, depending on the expected parameter type)
- Number system converter: just type in a number as a command (using the prefixes above except for hexadecimal where you should add a leading zero instead of none or $ sign to avoid conflicts with other commands). It will display the number's equivalents as a two byte value first in hexadecimal then in binary, decimal, ascii characters and finally as screen character codes
- Listing can be continued with scrolling from screen top up(except disassembly)/bottom down. Shift (or shift lock) pauses listings (it's useful with directories and file contents)
- All outputs can be redirected to printer by prefixing the command with an asterisk (*) character (for example: `ypu` selects a centronics compatible printer on the user port and `*d cef4 cff7` prints the disassembly)

Commands

Command	***Command name***	***Description***
£	List commands	Prints a list of valid commands
@[<command>]	Disk status and command	@ alone prints drive status @<command> executes disk command and drive prints status
A [<address> [<statement>]]	Assembly	Enter assembly statement
B [C|S <address>]	Breakpoint	B alone displays breakpoint Breakpoint clear Breakpoint set at address
C <src-address-from> <src-address-to> <dst-address-from>	Compare memory	Compare memory area src to dst
D M|[<address-from> [<address-to>]]	Disassembly	D <address-from> <address-to> disassembles the memory DM displays current settings DM$ turns on $ sign

Command	***Command name***	***Description***
		DM# turns off $ sign (default) DMI turns on illegal codes DMN turns off illegal codes (default)
E [<address>]	Edit screen	Edit screen In the menu: E-enter edit mode R-review (restore) contents S-store changes Q-quit from editor In edit mode press return to exit.
F <address-from> <address-to> <value>	Fill memory	Fills memory from <address-from> to <address-to> with byte <value>
G <address>	Go	Start execution
H <address-from> <address-to> <value>	Hunt for data	Search memory for specific data. data can be a string or any other byte sequence
I <address-from> <address-to>	Binary dump	Dump memory in binary format
J <address-from> <address-to>	Binary dump in multicolor mode	
K <address-from> <address-to>	ASCII (PETSCII) memory dump	
L [<drive>] "<filename>" [<startaddress>]	Load	
M <address-from> <address-to>	Memory dump (hexadecimal)	
N <address-from> <address-to> <search address-from> <search address-to> <address-offset>	New address (Program relocation)	
O [<address>]	Screen finder	
P [<address>]	Picture finder	
Q [<address>]	Character set finder	
R	Registers	
S [<drive>] "<filename>" <startaddress> <endaddress>	Save	

Command	*Command name*	*Description*
T	Transfer memory (copy data)	
U	Instructions	
V [<drive>] "<filename>" [<startaddress>]	Verify	
W [<address>]	Walk thru program (debug)	A simple debugger with real instruction execution (except jumps) in its real environment (memory config, IRq config, stack, etc.) Displays all CPU registers and the next instruction to be executed. Available keys: <space> - execute instruction <j> - execute jsr (real call!) <cr> - skip instruction <down> - skip instruction <stop> - quit debug mode <esc> - quit debug mode
X	Exit (reset)	
Y	Setup	Print settings Y Setting default memory YA Set RAM YO[<bank>] Set ROM <bank> YI[<drv>] Set drive <drv> Setting printer YP[<prt>] Set to serial <prt> YPU Set to Centronics printer on user port Set current drive YD[<drive>] Drive and interface scan YDS Expansion area scan YE – memory scan from $fd00-$feff YES – scan for known

Command	*Command name*	*Description*
		expansions
Z <address-from> <address-to>	Memory dump with screen codes	
;	Modify registers	
>	Modify memory (hexadecimal)	
<	Modify memory (ASCII/Screen)	
$[<filenamemask>]	Directory	
[	Modify memory (binary)	
]	Modify memory (multi-color binary)	
.	Enter assembly statement	
/ [<drive>] "<filename>" [<startaddress>]	Fast loader	Currently available only for serial and parallel 1541
= [<drive>] "<filename>"	Display file contents	
- <destination drive> "<filename>"	Copy file to another drive (unaccelerated)	

Memory used

Zeropage

```
ptr             = $be
ptr2            = $f1
ptr3            = $f3
```

Other RAM

```
dirname         = $0200 ; address to use as filename for directory command
keybuff         = $0201 ; keyboard buffer

memtemp         = $04eb ; temporary strorage for existing memory content
temp            = $04ec ; temporary storage
ctemp           = $04ed ; temporary storage for endaddress check
atemp           = $04ee ; temporary storage for register A
xtemp           = $04ef ; temporary storage for register X
ytemp           = $04f0 ; temporary storage for register Y

isbreak         = $04f1 ; breakpoint exists flag
brkbank         = $04f2 ; bank of breakpoint
brkaddr         = $04f3 ; address of breakpoint
brkdata         = $04f5 ; previous contents of breakpoint for restore

drive           = $04f6 ; current drive number. default is 8
printer         = $04f7 ; current printer device number. default is 4
defbank         = $04f8 ; default memory bank
wrtbank         = $04f9 ; memory bank for write operations
defrom          = $04fa ; default ROM bank
adrbank         = $04fb ; memory bank specified on destination address
```

```
monbank       = $04fc ; ROM bank where monitor is running (autodetected)
oribank       = $04fd ; originak ROM bank for switching back from IRQ.

monstack      = $053d ; stack pointer of monitor

shftflag      = $0543

mstrirq       = $054b ; original value of $fffe/$ffff

inendsgn      = $054d ; command line input end sign
bufflen       = $054e ; length of input buffer

irqjmp        = $054f ; original value of $0314/$0315
wasirq        = $0551 ; irq signal flag

regs          = $0552 ; register storage (same as built-in monitor)
pc            = regs
sr            = regs+2
ac            = regs+3
xr            = regs+4
yr            = regs+5
sp            = regs+6
oldirq        = regs+7
stackptr      = regs+9

brkrut        = $0609 ; RAM addresses for RAM resident routines (bank switching, etc.)
mstrirqr      = brkrut+14
irqrut        = mstrirqr+6

mode          = $07fe
mode2         = $07ff      ;0: CR Needs
                           ;1: Has FileName?
                           ;2: Load-Verify/Save
                           ;3: force word ass
                           ;4: rel-jump specified.
                           ;5: Must Specify address
                           ;6: $ Writing
                           ;7: Illegal Codes Disass
```