SimICE *Debugger for the SimCoupe Emulator* *© Simon Owen*

The **register panel** on the right shows the current system state. The first 7 lines of the panel show Z80 register values, with changed registers display in pink text. Below it are the current interrupt mode, and the interrupt state (EI or DI).

To the right of the interrupt state are 5 flags. These letters are visible when the corresponding interrupt type is active in the status port (249).

O = midi-out	**F** = frame	**I** = midi-in	**M** = mouse	**L** = line

ROM0/**ROM1**/**WPROT** show whether ROM0, ROM1 or RAM write-protection are active. White text is used for the active state, and dark grey inactive.

L/H/V/LE/HE are the current LMPR, HMPR, VMPR, LEPR and HEPR page numbers, with M showing the current screen mode (1-4).

Scan line:cycle shows the current TV line (0 to 311) and the current cycle position within the line 0 to 383.

T-diff shows the difference in tstates since the last view change. When single-stepping it represents the time for the last instruction, including CPU/ASIC contention. Stepping over a CALL will give the time for all code inside it, which provides a handy method of profiling code.

Keys for all views

A	enter new view address
D	disassembly view
T	text view
N	number view
G	graphics view
L	change LMPR page
H	change HMPR page
V	change VMPR page
M	change screen mode
Ctrl-0	toggle ROM0
Ctrl-1	toggle ROM1
Ctrl-2	toggle RAM write-protection
Ctrl-A	ex af,af'
Ctrl-D	ex de,hl
Ctrl-X	exx
Ctrl-I	toggle ei/di
Ctrl-T	toggle debugger transparency

Disassembly View

U	execute until condition is met
Keypad-7	single step 1 instruction
Keypad-8	step over instruction
Keypad-9	step out of function
Keypad-4	step 10 instructions
Keypad-5	step 100 instructions
Keypad-6	step 1000 instructions
Left/Right	scroll 1 byte
Up/Down	scroll 1 instruction
PgUp/PgDn	scroll 1 page
Ctrl-Left/Right	move PC by 1 byte
Ctrl-Up/Down	move PC by 1 instruction

Text/Number View

Up/Down	scroll by 1 line
Left/Right	scroll by 1 byte
PgUp/PgDn	scroll by 1 page

Graphics View

1/2/3/4	select screen mode
Up/Down	scroll by 1 line
Left/Right	scroll by 1 byte
Ctrl-Up/Down	zoom in/out
Ctrl-Left/Right	adjust column width by 1 byte
PgUp/PgDn	scroll by 1 column
Ctrl-PgUp/PgDn	scroll by 1 page

Single-stepping a HALT instruction will step into the interrupt handler, assuming interrupts are enabled. **Stepping over a HALT** will completely execute the handler, as if stepping over a call. Step-over also recognises JP/JR instructions, and will single-step to follow the jump rather than attempting to step over it.

To **return to the current execution point after browsing** other memory locations, press A to enter a new address and enter "pc" as the expression. Alternatively, single-step and the view will automatically **return to the next instruction**.

To aid to debugging, **conditional instructions** show whether or not the condition is met by the current flags. This makes it easy to determine whether a jump will be taken, with an arrow indicating its direction.

Double-clicking on an instruction in disassembly view will **set an execution breakpoint** for that address (no matter where it's paged in memory). There's currently no way to list existing breakpoints, or set explicit new ones.

The most powerful feature in the current implementation is the 'U' command, which **executes until an expression is met**. You can create complex expressions using the following:

Operators

Unary	+ - ~ ! =	**Comparison**	== != < > <= >=
Binary arithmetic	+ - * / %	**Bitwise arithmetic**	& | ^
Logical	&& ||	**Bitwise shift**	<< >>

Symbols

Single registers	a f b c d e h l i r ixh ixl iyh iyl
Double registers	af bc de hl af' bc' de' hl' ix iy sp pc
Interrupts	ei di iff1 iff2 im
Paging	lmpr hmpr vmpr mode lepr hepr rom0 rom1 wprot
Display	dline sline lcycles

Functions

PEEK <addr>	8-bit lookup in currently paged RAM
DPEEK <addr>	16-bit lookup in currently paged RAM
IN <port>	non-zero if previous instruction accessed the port
OUT <port>	non-zero if previous instruction accessed the port

The '=' unary **operator** has a special use in expressions. Its operand is evaluated immediately, and the value inserted in the expression instead of the operand itself. The first example below shows why this can be useful.

Example expressions

Break when the current value of HL changes	hl != =hl
Break at the next HALT instruction	peek pc == 0x76
Break when a floppy command is written	out 224
Break when screen mode 3 is selected	mode == 3
Break when 12345 is top of stack	dpeek sp == 12345
Break when the raster is drawing screen line 0	sline == 0
Break when A, B and IXl are equal	(a == b) && (b == IXl)

Execute Until breakpoints are only temporary, and cleared when the debugger is next activated, regardless of whether they were triggered. This also applies to other simple breakpoints, such as step-out and step-over.

Values are displayed in hex in both disassembly and number modes, but values used in inputs and expressions can be in many different bases. The following number formats can be used:

Decimal	12345
Character	"a" or 'a'
Binary	%10101100 or 10101100b
Hexadecimal	0x1234 or 1234h or $1234 or &1234 or #1234

Octal is not supported, so leading zeroes have no special meaning.