Core War/Redcode

Core War programs are written in a programming language known as Redcode. Redcode does not have access to direct input or output.

Opcodes

Each opcode in Corewar contains three components: The instruction, register A, and register B. In the core, each memory section contains exactly one opcode, and provides no independent numbers.

The registers A and B are relative addresses, described in the Address modes below.

Opcode	Description
DAT A,B	Data; executing this instruction kills the task.
MOV #A,B	If one is immediate, copied immediate number to B.
MOV A,B	Moves instruction at A to B. However, if A is immediate, copies the immediate number to B.
ADD A,B	Adds A to B.
SUB A,B	Subtracts A from B.
MUL A,B	Multiplies A and B to produce the result.
DIV A,B	Divides B by A. Division by zero kills the process.
MOD A,B	Modulus of B in A. Division by zero kills the process.
JMP A,B	Jumpts to A.
JMZ A B	If *B is 0, jumps to A.
JMN A,B	If *B is not 0, jumps to A.
DJN A,B	Decreases *b. If non zero, jumps to A.
CMP A,B	If *a == *B (or if immediate, #A == *B), skips an instruction
SPL A,B	Creates a new task, which starts at B. Newer task starts first.
SEQ A,B	Skips past the next instruction is A and B are equal.
SNE A,B	Skips past the next instruction is A and B are not-equal.
SLT A,B	Skips if *A < *B. (88 only)
XCH A,B	At A, swaps A and B. (extended only)
PCT A,B	At A, Protects operands from changing, until an instruction is written to that address.
NOP A,B	Has no special effect (although operands are still evaluates.)
STP A,B	Stores A into P-Space at location specified by B.
LDP A,B	Retrieves data from P-Space at location specified by B into A.

Address modes

Each opcode listed above contains two registers. These registers are composed of both a number and an address mode:

#	Immediate. The number is directly in the opcode.
$ (or none)	Direct. The opcode points to a cell relative to the current cell.
@	Indirect. The opcode points to a cell relative to the current cell. That cell's B value is added o the indirect pointer, to provide the target.
<	Indirect, but the intermediate register is decreased before use.
>	Indirect, the intermediate register is increased after use.
*	A-field Indirect. The opcode points to a cell relative to the current cell. That cell's A value is added to the indirect pointer, to provide the target.
{	A-field Indirect, but the intermediate register is decreased before use.
}	A-field Indirect, the intermediate register is increased after use.

Special commands

END	Stops compilation, further lines are treated as comments.
ORG	Takes one parameter, which identifies the start location.
PIN	Specifies P-Space identifier. If equal, the two programs share P-Space.
<label> EQU <A>	Replaces all instances of <label> with <A>.

Instruction modifiers

.A	A -> A
.B	B -> B
.AB	A -> B
.BA	B -> A
.F	A->A and B->B
.X	A->B and B->A
.I	Entire instruction.

P-Space

P-Space is a private storage used by programs across multiple runs of a program, that cannot be directly accessed by the opponent. However, attacks by programs can trick other programs into corrupting the P-Space region

In the opcode listing above, there are two instructions that can read and write to P-Space: STP and LDP.

Article Sources and Contributors

Core War/Redcode *Source*: http://en.wikibooks.org/w/index.php?oldid=2064371 *Contributors*: Avicennasis, Sigma 7

License