No Operation Instructions

Nop-A, Nop-B, Nop-C, Nop-D

These instructions do not perform any action when executed, but can be used as parameters for other functions. Typically, these are used to specify a particular stack to use for an operation, but they can also be used as a search pattern (see **Search**) or a label for a particular memory space (see **SetMemory**). Note that most operations that use 2 Nop instructions to specify which stacks to use can also be parameterized by a single Nop, in which case the second parameter is assigned the next stack in sequence (e.g., C->D, D->A, etc.).

Single Argument Math/Data Instructions

Val-Shift-R This instruction shifts the value at the top of the specified stack (via Nop-instruction, or Stack-B by default) over 1 bit to the right, effectively dividing the value by 2 and rounding down.

Val-Shift-L This instruction shifts the value at the top of the specified stack (via Nop-instruction, or Stack-B by default) over 1 bit to the left, effectively multiplying the value by 2 and ignoring any potential overflow.

Val-Inc This instruction increments the value at the top of B stack by 1. This instruction is destructive (I.e., it pops the old value off the stack). This instruction is optionally parameterized by a single Nop.

Val-Dec This instruction decrements the value at the top of B stack by 1. This instruction is destructive (I.e., it pops the old value off the stack). This instruction is optionally parameterized by a single Nop.

Val-Delete This instruction pops off a value from the B stack. This instruction may be modified by a single Nop instruction, which determines from which stack to remove a value.

Double Argument Math/Data Instructions

Val-Nand This instruction computes the bitwise NAND operation on the values at the top of the B and C stacks (each of which are 32-bit numbers). The result of this operation is pushed onto the top of the B stack. The stacks used may be parameterized by 1 or 2 optional Nop instructions. This is the only logic operation provided in the standard Avida instruction set.

Val-Add This instruction pushes the sum of values from the top of the B and C stacks onto the B stack. This instruction is optionally parameterized by 1 or 2 Nop instructions.

Val-Sub This instruction pushes the result of subtracting the values from the top of the B and C stacks (B - C) onto the B stack. This instruction is optionally parameterized by 1 or 2 Nop instructions.

Val-Mult This instruction pushes the result of multiplying the values from the top of the B and C stacks onto the B stack. This instruction is optionally parameterized by 1 or 2 Nop instructions.

Val-Div This instruction pushes the result of integer division of the value from the top of the B stack by the value at the top of the C stack (B/C). This instruction is optionally parameterized by 1 or 2 Nop instructions. The instruction fails and does not return a value if there is an arithmetic error caused by the division.

Val-Mod This instruction pushes the remainder from integer division between the value at the top of the B stack by the value at the top of the C stack (B%C). This instruction is optionally parameterized by 1 or 2 Nop instructions. The instruction fails and does not return a value if there is an arithmetic error caused by the division.

Val-Copy This instruction pushes a copy of the value at the top of the B stack onto the B stack. This instruction is optionally parameterized by 1 or 2 Nop instructions, where the first Nop specifies the source stack, and the second Nop specifies the destination stack. In this case, if only one Nop instruction is used, the value is copied and pushed onto the same stack as the source.

Push-Next This instruction pops a value off of the A stack and pushes it onto the next stack in sequence (the B stack). This instruction is optionally parameterized by 1 or 2 Nop instructions, where the first Nop specifies the source stack, and the second Nop specifies the destination stack.

Push-Prev This instruction pops a value off of the B stack and pushes it onto the previous stack in sequence (the A stack). This instruction is optionally parameterized by 1 or 2 Nop instructions, where the first Nop specifies the source stack, and the second Nop specifies the destination stack.

Push-Comp This instruction pops a value off of the B stack and pushes it onto the complement stack (I.e., the stack 2 over in sequence), in this case, the D stack. This instruction is optionally parameterized by 1 or 2 Nop instructions, where the first Nop specifies the source stack, and the second Nop specifies the destination stack.

Flow Control Operations

If-Equal If the value on the top of the A stack is equal to the value on the top of the B stack, the next instruction (after any modifying Nop instructions) will be executed, otherwise it will be skipped. This instruction is optionally parameterized by 1 or 2 Nop instructions, which specify which stacks are compared.

If-Not-Equal If the value on the top of the A stack is not equal to the value on the top of the B stack, the next instruction (after any modifying Nop instructions) will be executed, otherwise it will be skipped. This instruction is optionally parameterized by 1 or 2 Nop instructions, which specify which stacks are compared.

If-Less If the value on the top of the A stack is less than the value on the top of the B stack, the next instruction (after any modifying Nop instructions) will be executed, otherwise it will be skipped. This instruction is optionally parameterized by 1 or 2 Nop instructions, which specify which stacks are compared.

If-Greater If the value on the top of the A stack is greater than the value on the top of the B stack, the next instruction (after any modifying Nop instructions) will be executed, otherwise it will be skipped. This instruction is optionally parameterized by 1 or 2 Nop instructions, which specify which stacks are compared.

Head-Push This instruction pushes the position of the IP head onto the B stack. This instruction can be modified by 1 or 2 Nop instructions. The first Nop specifies which head

(READ, WRITE, FLOW, IP) is used, and the second Nop specifies on which stack the position is pushed. Note that this instruction differs from Inst-Read in that it pushes the location of the specified head rather than the value at that position.

Head-Pop This instruction pops a position off the B stack and moves the IP head to that position. This instruction can be modified by 1 or 2 Nop instructions. The first Nop specifies which head (READ, WRITE, FLOW, IP) is moved, and the second Nop specifies from which stack the position is popped.

Head-Move This instruction moves the IP head to the position pointed to by they FLOW head. This instruction is optionally parameterized by one Nop instruction that specifies which head to move to the position pointed to by the FLOW head.

Biological Operations

SetMemory This instruction selects which memory space is active (I.e., which memory space the flow head is pointing to). The memory spaces can be labeled with up to 4 Nop instructions. If no Nop instructions are given, the memory space containing the currently executing organism is selected. If a Nop label is provided that does not yet exist, a new memory space is created.

Inst-Read This instruction pushes the instruction pointed to by the READ head onto the A stack and advances the READ head by one position. This instruction is optionally parameterized by 1 or 2 Nop instructions. The first Nop specifies which head (READ, WRITE, FLOW, IP) is used, and the second Nop specifies on which stack the instruction is stored.

Inst-Write This instruction pops the instruction on the A stack and writes it into the memory location pointed to by the WRITE head, and and advances the WRITE head by one position. This instruction is optionally parameterized by 1 or 2 Nop instructions. The first Nop specifies which head (READ, WRITE, FLOW, IP) is used, and the second Nop specifies from which stack the instruction is read.

Divide-Erase This instruction attempts to divide off a finished offspring copy. It uses whatever memory space the WRITE head is currently pointing to as the offspring's genome. If this instruction fails, the memory space is cleared and the organisms heads are reset.

Inject This instruction acts similarly to Divide-Erase, but instead of dividing off a "free-living" organism, it attempts to infect an organism with a parasite genome contained in the memory space pointed to by the WRITE head.

Input/Output and Sensory Operations

IO This instruction handles the input and output components of Avida. It will output the value at the top of the B stack to the environment, checking if this value is a logical function of any of the random numbers provided to the organism during its lifetime. It also pushes a randomly generated value from the environment (the input component) onto the B stack. This instruction is optionally parameterized by 1 or 2 Nop instructions. The first Nop instruction specifies the input stack, and the second specifies the output stack. If only one Nop is used, the same stack is used for both the input and output values.

Search This instruction will read in the sequence of Nop instructions that follows it, and search for the location of a *complement* template (e.g., BCD -> DAB) in the organism's genome. The complement is the label where each Nop is shifted over in sequence by 2, such that the complement of the complement recovers the original label. If one is not found, a value of 0 will be pushed to the top of the B stack. If a complement is found, the distance between the current instruction being executed and the beginning of the compliment will be pushed onto the B stack, the size of the label will be pushed on the A stack, and the FLOW head will be moved to the end of the compliment template.