Lab # 3

# OBJECTive

Define the memory architecture of VVM

# THEORY

### BRP *nn*

**Branch if Positive or Zero (8*nn*) [BRP *nn*]**This is a conditional branch instruction. If the value in the Accumulator Register is positive or zero, then the Program Counter Register is replaced by the operand value *nn*. The result is that the next instruction to be executed will be taken from address *nn* rather than from the next sequential address. Otherwise (Accumulator < 0), the Program Counter Register is incremented by one, and the next sequential instruction is executed.

### BRZ *nn*

### Branch if Zero (7*nn*) [BRZ *nn*]This is a conditional branch instruction. If the value in the Accumulator Register is zero, then the Program Counter Register is replaced by the operand value *nn*. The result is that the next instruction to be executed will be taken from address *nn* rather than from the next sequential address. Otherwise (Accumulator <> 0), the Program Counter Register is incremented by one, and the next sequential instruction is executed.

### BR *nn*

**Branch (6*nn*) [BR *nn*] (or [BRU *nn*] or [JMP *nn*])**This is an unconditional branch instruction. The current value of the Program Counter Register is replaced by the operand value *nn*. The result is that the next instruction to be executed will be taken from address *nn* rather than from the next sequential address. The value of the Program Counter Register is not incremented with this instruction.

**VVM Program # 3**

**Simple looping example.**

In Input A

sto 99 Store A

**Equivalent to the following BASIC program:**

**INPUT A**

**DO WHILE A > 0**

**PRINT A**

**INPUT A**

**LOOP**

**END**

brp 04 [02] If A >= 0 then skip next

br 10 Jump out of loop (Value < 0)

brz 10 [04] If A = 0 jump out of loop

lda 99 Load value of A (don't need to)

out Print A

in Input new A

sto 99 Store new value of A

br 02 Jump to top of loop

hlt [10] Done

**VVM Program example# 4**

**Sample program to print the square of any integer in the range 1-31.**

in Input value to be squared

sto 99 Store input at 99

lda 98 Load current sum (top of loop)

add 99 Add value to sum

sto 98 Store the sum

lda 97 Load current index

add 96 Add 1 to index

sto 97 Store new index value

sub 99 Subtract value from index

brz 11 Jump out if index = value

br 02 Do it again (bottom of loop)

lda 98 Done looping - load the sum

out Display the result

hlt Halt (end of program)

// Data used by program follows

\*96 Resume loading at address 96

dat 001 Constant for counting

dat 000 Initial index value

dat 000 Initial sum

# Assignment

Write a VVM program which take an integer input and display table of that integer.