7-2

CALL 1110 => SP := SP-1 ; M[SP] := PC ; PC := X

RETIV 1111 1000 0- => PC := M[SP] ; SP := SP + 1

PUSH 1111 0100 0- => SP := SP-1 ; M[SP] := AC

POP 1111 0110 0- => AC := M[SP] ; SP := SP + 1

PSHI 1111 0000 0- => SP := SP-1 ; M[SP] := M[AC]

POPI 1111 0010 0- => M[AC] := M[SP] ; SP := BP + 1

SWAP 1111 1010 0- => swap AC and SP

INSP 1111 1100 8y => SP := SP + y (0 ≤ y ≤ 255)

DESP 1111 1110 8y => SP := SP - y "

HALT 1111 1111 0- => stops machine

MAC-1 () initial value of AC, PC, SP = 0 = 0 = 4095 {

while true {

I = M[PC] ← FETCH

switch (I) { ← DECODE

case LODD : AC = M[x] ; continue } ← EXECUTE

...

case HALT : return ;

}

}

}

Assume that "A" is a location ↳ Ptr to ADATA ↙ N values and "B" is a location ↳ Ptr to BDATA ↙ N values and "N" is a location ↙ number

Purpose: Push A[i] + B[i] onto the stack for $i \in [0, N]$
Leave AC as the total of all A & B

Purpose: Compute all the Fibonacci numbers the MIC-1 can
↳ Save them to the stack

```
STOL 0          // AC = 0  => m[SP] = Fib(0) = 0
LOCO 1          //         => AC = 1
STOL 1          // AC = 1  => m[SP+1] = Fib(1) = 1
loop: ADDL 0    // AC = Fib(n+1)  m[SP] = Fib(n)  => AC = Fib(n+2)
      STOL 2    //  m[SP+2] = Fib(n+2)
      INSP 1    // AC = Fib(n+1)  m[SP] = Fib(n)
      JPOS loop:
```

n' = n+1

Instruction encoding RISC / MIPS