Mini CPU

Machine model
- all registers are 8 bits
- there are 16 general purpose registers (i.e. R0 thru R15)
- there are two special purpose registers (accumulator AC and program counter PC)
- most instructions are 1 byte, some special instructions are 2 bytes
- there are two I/O ports, one for input, one for output
- there is no stack
- there are no interrupts

Memory model
- Instruction RAM is 256 bytes
- Data RAM is 256 bytes

Instruction Format

```
7 6 5 4  3 2 1 0
_ _ _ _  _ _ _ _
```

Bits 7-6 - specify instruction type
Bits 5-4 - specify instruction within the type
Bits 3-0 - specify register or other function code

Bits 7-5

00xx - special

```
0000 mmmm                       misc instructions
0001 rrrr       CPR Rn          compare acc with register       - if A == R -> Z=1, else Z=0
0010 rrrr       CPM Rn          compare acc with data memory    - if A == D[R] -> Z=1, else Z=0
0011 bbbb                       branch instructions

bbbb codes
0001            BRZ aa          branch if zero to address       - if Z=1 -> I[PC+1] -> PC
```

```
0010            BNZ aa          branch if not zero to address   - if Z=0 -> I[PC+1] -> PC
0100            BRC aa          branch if carry to address      - if C=1 -> I[PC+1] -> PC
1000            BNC aa          branch if not carry to address  - if C=0 -> I[PC+1] -> PC
....
1111            BRU aa          unconditional branch to address - I[PC+1] -> PC

mmmm codes
0000            NOP             no operation                    - no action
....
0100            INP             load input port to acc          - <in> -> A
0110            OUT             send acc to output port         - A -> <out>
....
1000            INV             invert acc                      - A' -> A
1010            LDA dd          load acc with immediate value   - I[PC+1] -> A
....
1100            RST             reset cpu                       - 0 -> PC, 0 -> AC
1110            HLT             halt                            - PC - 1 -> PC


01xx - transfer

0100 rrrr       LDM Rn          load acc from data memory       - D[R] -> A
0101 rrrr       STM Rn          store acc to data memory        - A -> D[R]
0110 rrrr       MRA Rn          move register to acc            - R -> A
0111 rrrr       MAR Rn          move acc to register            - A -> R


10xx - math

1000 rrrr       ADD Rn          add register to acc             - A + R -> A
1001 rrrr       SUB Rn          sub register from acc           - A - R -> A
1010 rrrr       INC Rn          increment register              - R + 1 -> R
1011 rrrr       DEC Rn          decrement register              - R - 1 -> R
```

```
11xx - logic

1100 rrrr       AND Rn          and acc with register           - A and R -> A
1101 rrrr       ORR Rn          or acc with register            - A or R -> A
1110 rrrr       XOR Rn          xor acc with register           - A xor R -> A
1111 ssss                       shift instructions

ssss codes
....
0001            LSR             logical shift acc right         - A >> 1 -> A
0010            LSL             logical shift acc left          - A << 1 -> A
....

Examples:

# add numbers 1 thru 5 send result to output
:start  NOP
        LDA #6
        MAR R5          ;init R5 to loop limit
        LDA #0
        MAR R3          ;init R3 to 0 (use this as sum)
        LDA #1
        MAR R1          ;init R1 to 1 (use this as current number)
        NOP
:loop   MRA R3          ;get the sum
        ADD R1          ;add number to sum
        MAR R3          ;save the sum
        INC R1          ;increment to next number
        MRA R1
        CPR R5          ;are we at the loop limit?
        BNZ &loop       ;no, repeat loop
        NOP             ;yes, we're done
        MRA R3
        OUT             ;output sum as result
        HLT
```