TOY CPU: Project 5

1. Objective

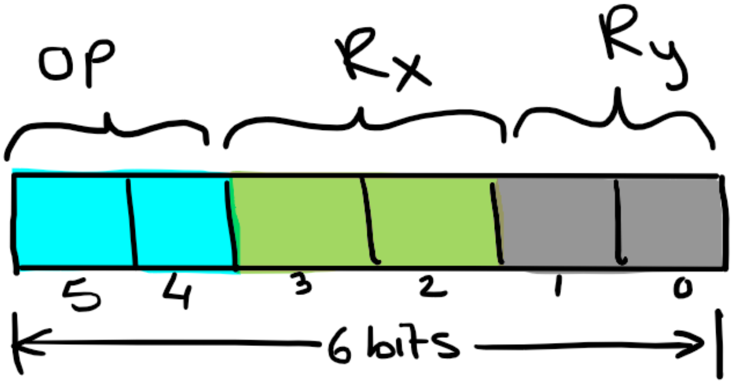
We will create a simple CPU with a simple instruction set architecture.

2. Simple TOY ISA

- The instruction for this ISA is 6 bits long and supports 4 operations.
- The CPU has 4 general purpose 16-bit registers
- The CPU will interface with a 32K ROM and 32K memory chip
    - Will require 15 bits for addresses

2.1 Instruction

The instruction look like so:



The op codes are:

- 00 : read
- 01 : write
- 10 : add
- 11 : jump

Rx and Ry are the operand registers.

- 00 refers to register 0
- 01 refers to register 1
- 10 refers to register 2
- 11 refers to register 3

The instructions behave as follows:

read Rx Ry ⇒ Rx = Memory[Ry]
write Rx Ry ⇒ Memory[Rx] = Ry
add Rx Ry ⇒ Rx = Rx + Ry
jump Rx Ry ⇒ Jump to ROM[Rx] if Ry == 0

2.2 Semantics

It is important to note that the certain bits in the instruction have meaning when it comes to what the CPU should do. For example, the first two bits represent the type of instruction to be performed. This can be taken in to consideration when implementing the chip.

3. CPU Datapaths

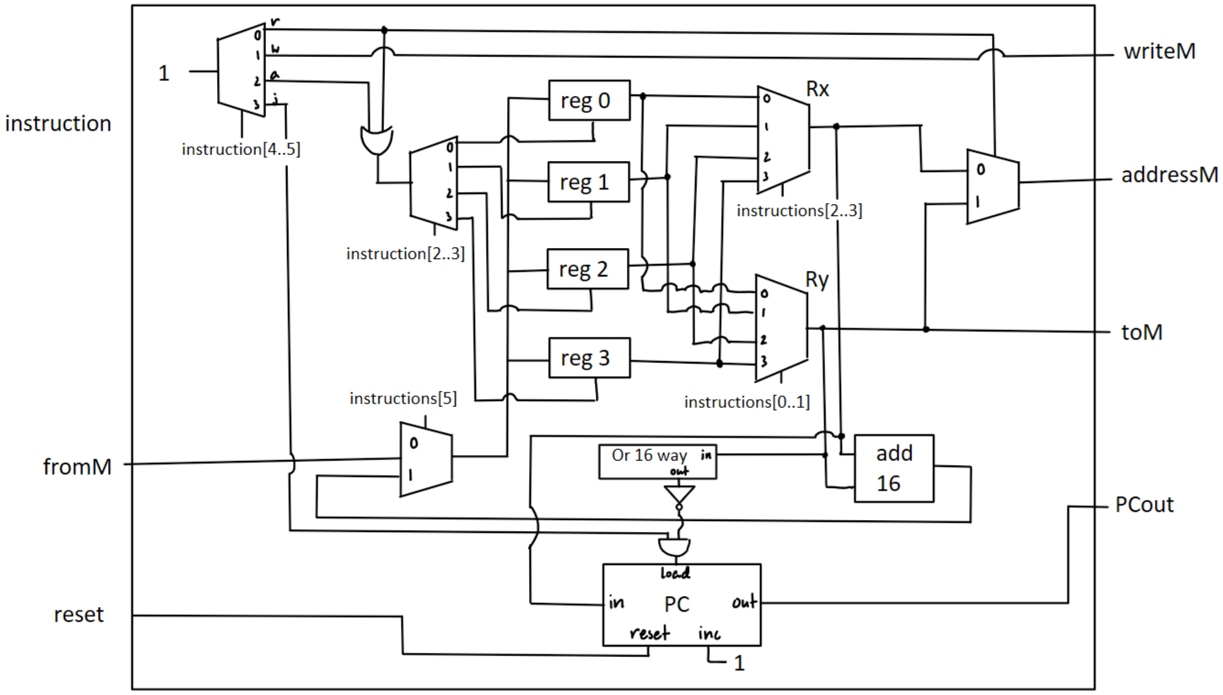
3.1 Inputs and Outputs

The CPU will have the following inputs and outputs:

```
inputs:        Instruction[6], fromM[16], Reset
outputs:       toM[16], addressM[15], writeM, PCout[15]
```

Instruction - the 6 bit instruction described in section 2.
fromM - the value read from the memory.
Reset - signals to reset the PC to 0

toM - the data being sent to be written to memory
addressM - where in memory to place the data in toM
writeM - whether or not to write to memory.



Cpu.hdl:

```
CHIP CPU {
   IN Instruction[6], fromM[16], Reset;
   OUT toM[16], addressM[15], writeM, PCout[15];

   PARTS:

}
```

Submission and Testing

Test the hdl file on the Hardware simulator and submit on Gradescope.