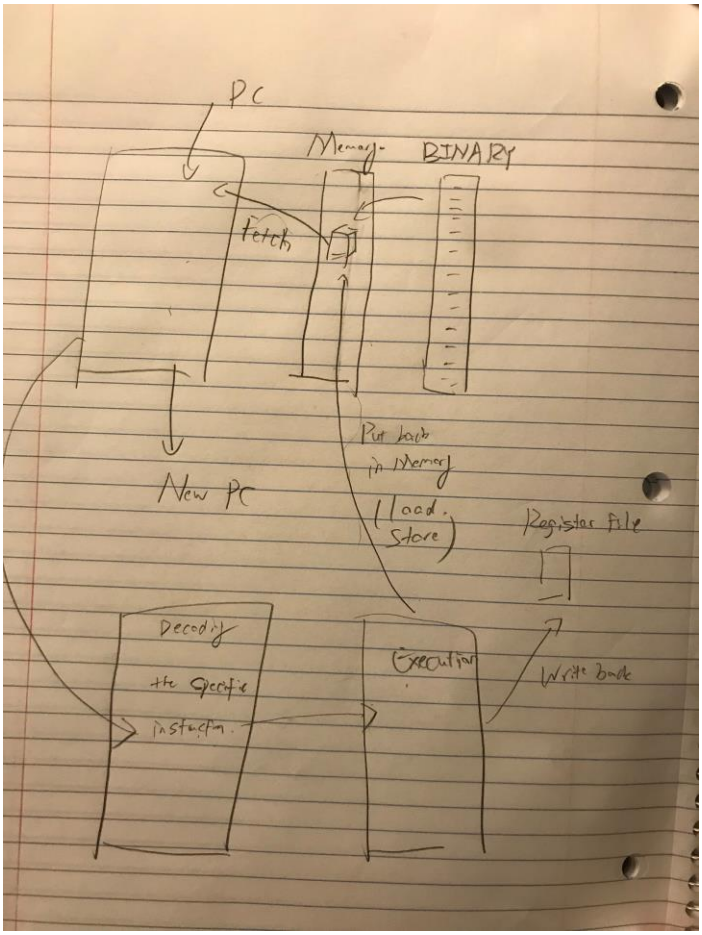
**Design Doc:** Detailing the structural and logical design of your program

**DIAGRAM**: interface, flow chart




**Details of logical design**

Our program will implement the single instruction within the following 5 stages.

- **Fetch**
  - Read instruction content from the Memory by using the given PC address
  - Compute the new pc address by “PC + 4” , written back to “*new_pc”
- **Decode**
    - The IF/ID pipeline register supplies immediate field, register numbers to read two registers. We are going to store the 3 values in the ID/EX pipeline registers for future use (later cycle).
  - Determine the instruction type by the “opcode” [6:0]
  - Split the whole instruction code into different pieces by the function format
  - Determine the specific function by the “function number” [9:7]
  - Read the data we need from the memory.
  - Use Switch-case function to direct to different sub-functions in Execute Stage.
- **Execute**
  - Sub-functions that implement the execution for different functions

        - For testing and scalability purposes, all functions should be independent
        - Should be a lot !!!
        - As simple as possible
    - Each sub-function should call memory_write(), or register_write(), or *new_pc
- **Memory**
    - Write the data in the memory depending on the instruction requirement
        - Not necessary if not required
- **Write**
    - Write the data in the registers depending on the instruction requirement
        - Not necessary if not required

**Things to consider**

- Pipeline
    - For better supporting the following program assignments, all the stages should be written in different functions so that we can call them individually.
- Comment
    - Each sub-function should contain a brief comment, indicating what the function do, what parameters the function need, and what the result of the implementation.
- Interface
    - Consider how to reserve space and set interfaces for stage registers in the next assignment

**Task assignment**

- Hang Yuan
    - The basic structure of the program
    - No. 20-35 sub-functions in execution stage
    - Debugging
- Zhewei Wang
    - Memory stage
    - No. 36-51 sub-functions in execution stage
    - Debugging
    - Testing
- Andrew Tsai
    - Register stage
    - No. 1-19 sub-functions in execution stage
    - Debugging