YOUR Group Details:

Group No:	16
Group Members (*Student IDs*):	Suhana Sajid (20469467) Phoebe Chan Mun Zheng (20312783)

Justification of YOUR Circuitry Diagram Design:

(**NOTE**: No more than 300 words)

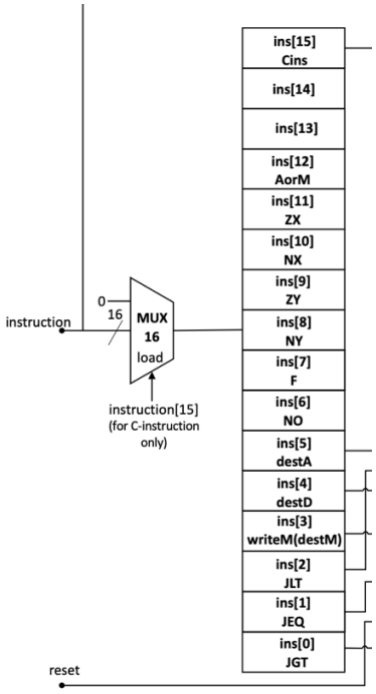

Figure 1 Instruction decoding

- A Mux16 gate is used to determine the type of instruction.
- If instruction[15] = 1, the instruction is a C instruction, else an A instruction.
- Only C instruction will be decoded because its bits are used to control all the processes in the CPU such as the ALU, PC, etc.

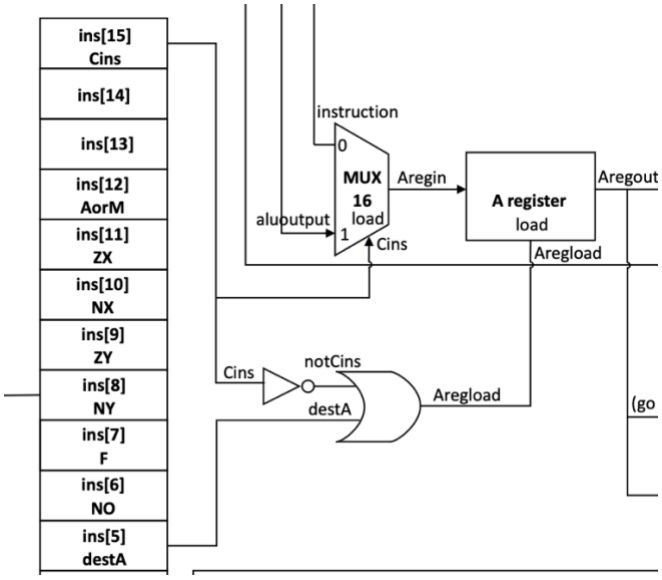

Figure 2 A register and input multiplexer

- A NOT gate is used to negate instruction[15] because if the instruction is not a C instruction, it is an A instruction.
- It is followed by an OR gate because the A register is only updated if it is either an A instruction (not C instruction) or instruction[5] is set. The OR gate determines the A register's load value.

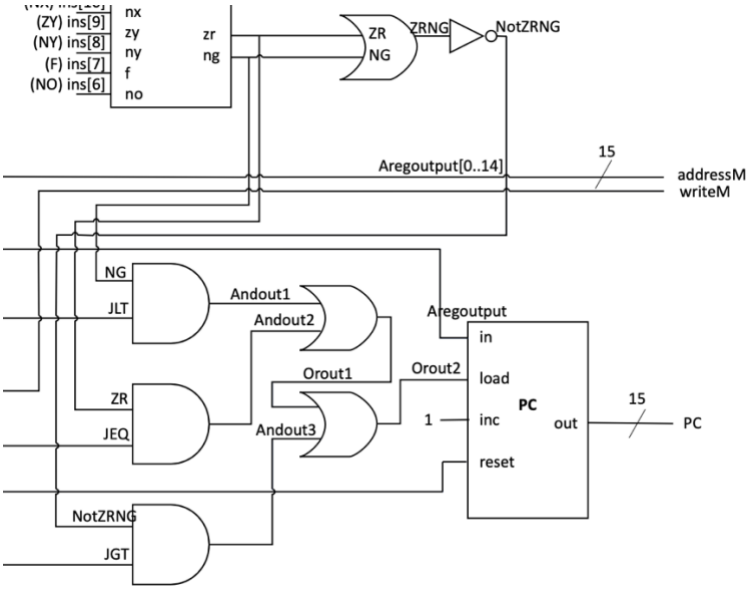

Figure 3 PC with jump test

- Instruction[0] to instruction[2] and the ZR and NG flags of the ALU are used to determine if the PC's load is set.
- If NG = 1 AND the instruction to jump if the result is less than zero, instruction[0] = 1, the PC's load is set.

- If ZR = 1 AND the instruction to jump if the result is equal to zero, instruction[1] = 1, the PC's load is set.
- If the ALU's result is not zero OR not negative, it is positive.
- This justifies the usage of the OR gate followed by a NOT gate as shown in Figure 3.
- If the ALU's result is positive AND the instruction to jump if the result is greater than zero, instruction[2] = 1, the PC's load is set.
- The only possibility that the PC's load is not set is when instruction[0] to instruction[2] are all zeros. Justifying the use of AND here.
- If any of these conditions (calculated using AND gates) is true, the PC load is set, justifying the use of OR gates.