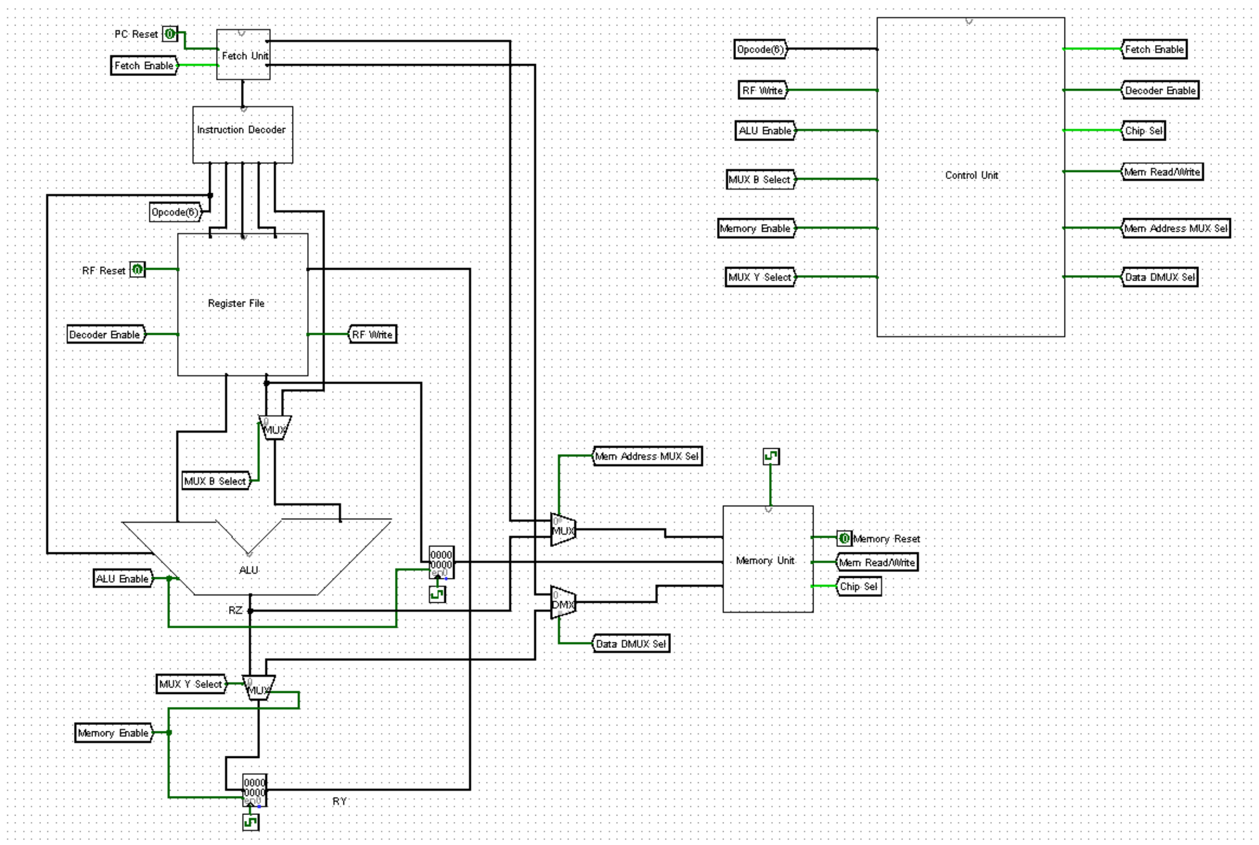
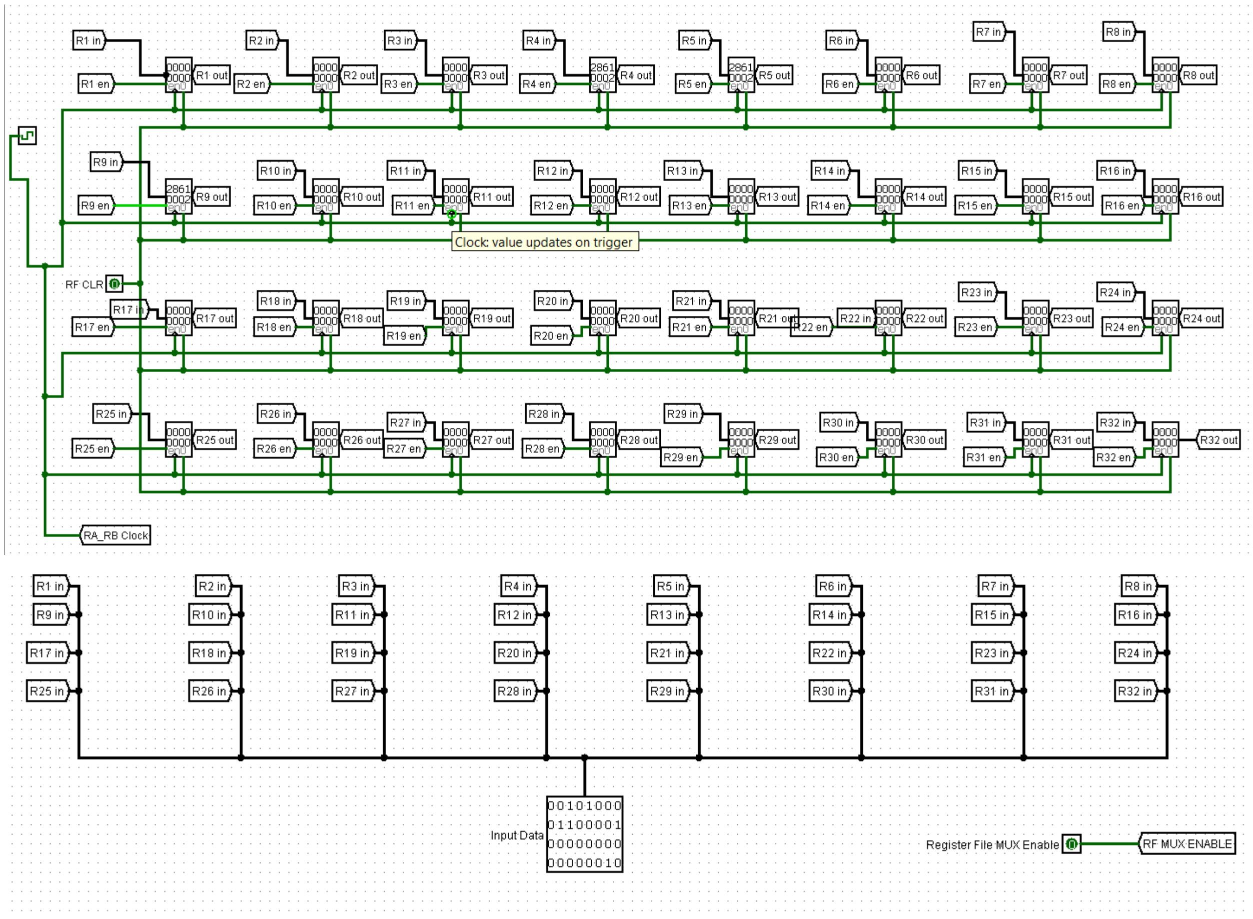
LOGISIM CIRCUIT IMPLEMENTATION DETAILS

5-STAGE PIPELINE OF 32-BIT PROCESSOR



REGISTER FILE IMPLEMENTATION(SHOWING 32 REGISTERS, INPUTS AND MUX ENABLE)

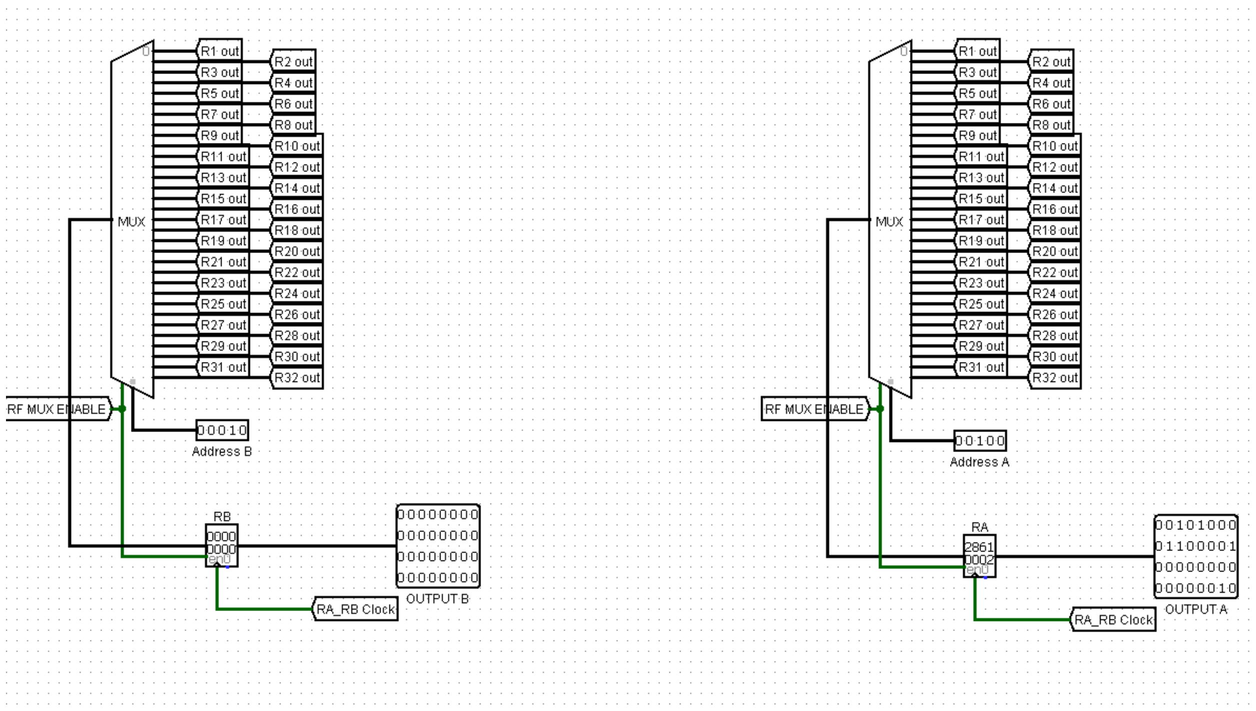
1)



- We have included 32 registers in the register file with a common clock for each one of them.
- All the inputs to the registers are fed by the common pin named Input Data.
- The Register File MUX Enable enables the MUX to select RA and RB based on the 5 bit address provided.

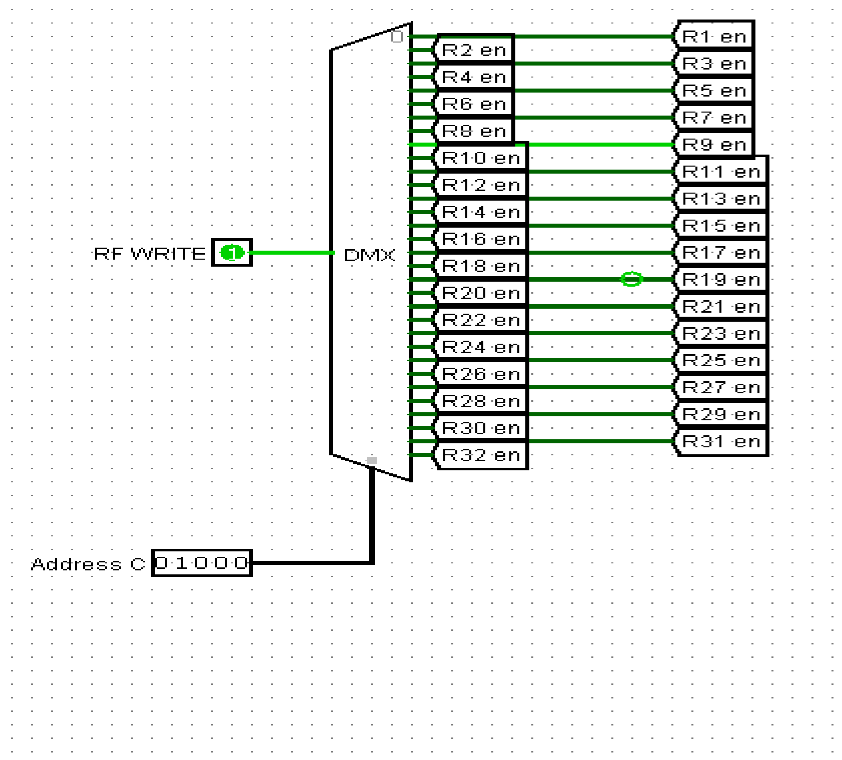
REGISTER FILE IMPLEMENTATION(SHOWING MUX and DEMUX)

2)



- The MUX on the left selects one out of 32 registers based on the address and stores the contents of it into the register B.
- The same phenomenon occurs for the right MUX, except it is for the register RA.
- The RA_RB clock is the clock tunnel which is common to the entire register file cirucit.

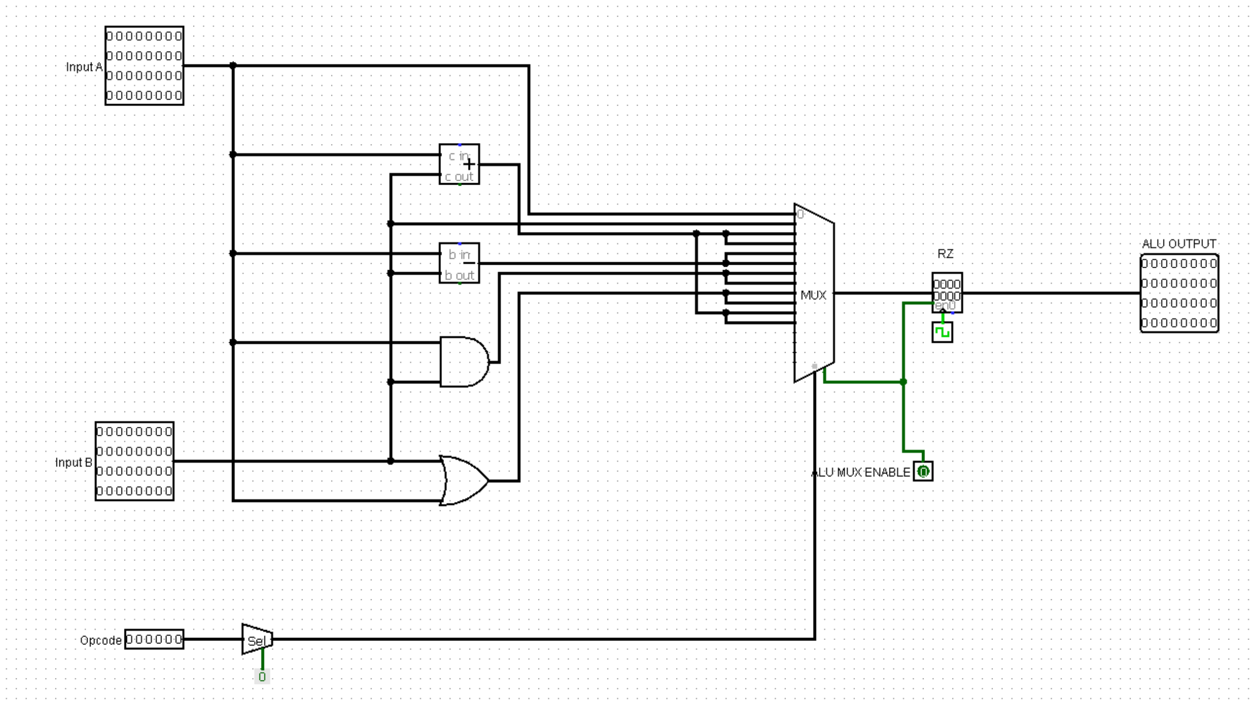
3)



- The DEMUX above decides on which register the writeback is to be done (when RF WRITE=1). This information is provided by the 5 bit address **Address C.**

—---

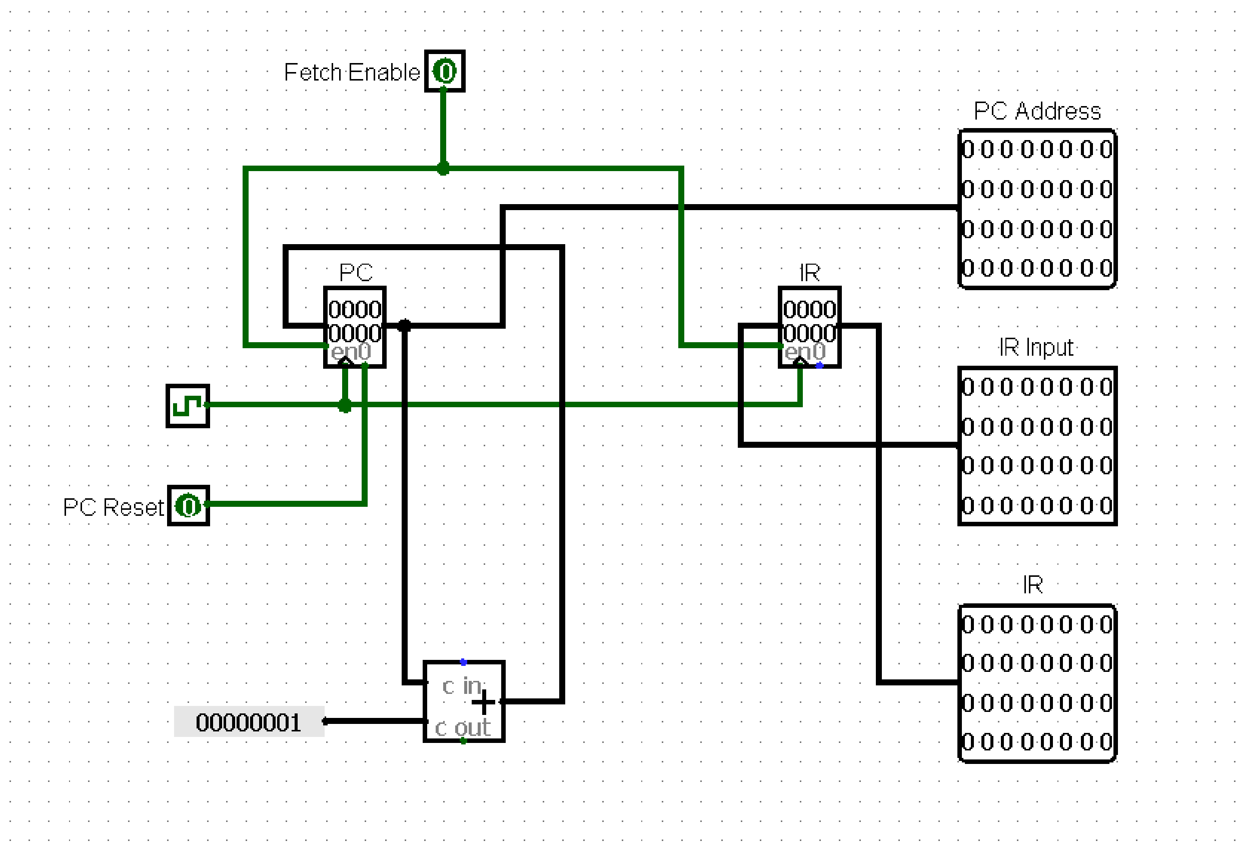
ARITHMETIC LOGIC UNIT (ALU) IMPLEMENTATION



- The ALU takes two Inputs A and B. The MUX has 16 inputs(as opcode is of 4 bits if we neglect the first 2 don't care bits) and 1 output. The inputs of the MUX are interpreted as follows:
  I0:MOV
  I1 :MVI
  I2 :ADD
  I3 :ADI
  I4 :SUB
  I5 :SUI
  I6 :AND
  I7 :ANI
  I8 :OR
  I9 :ORI
  I10 :LOAD
  I11 :STORE

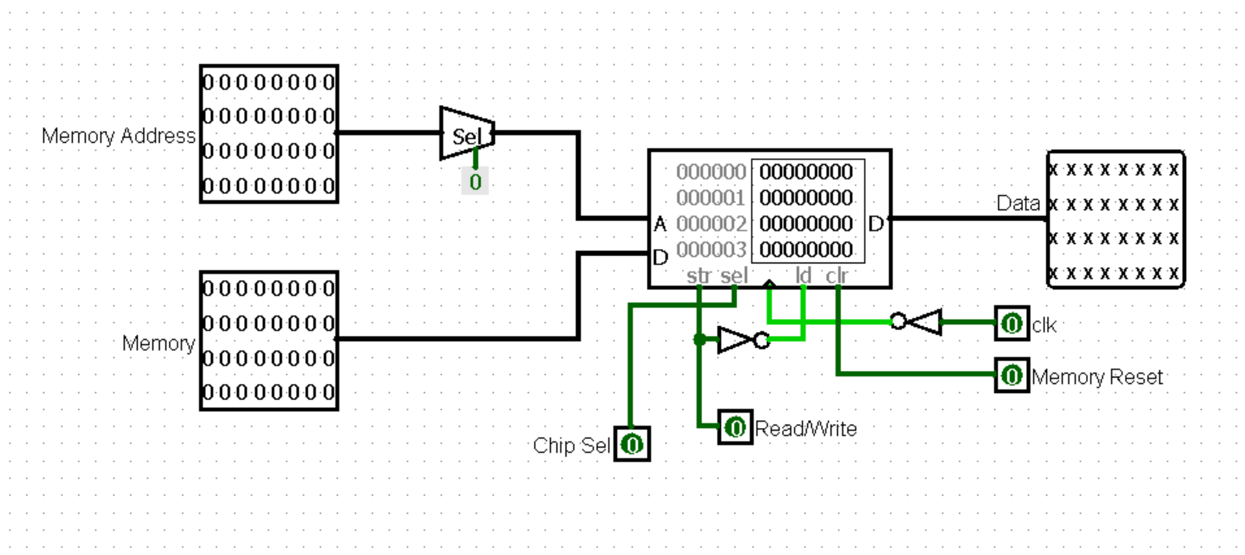
The computed output is stored in the register RZ which may be used in the coming stages of the pipeline.

FETCH UNIT IMPLEMENTATION



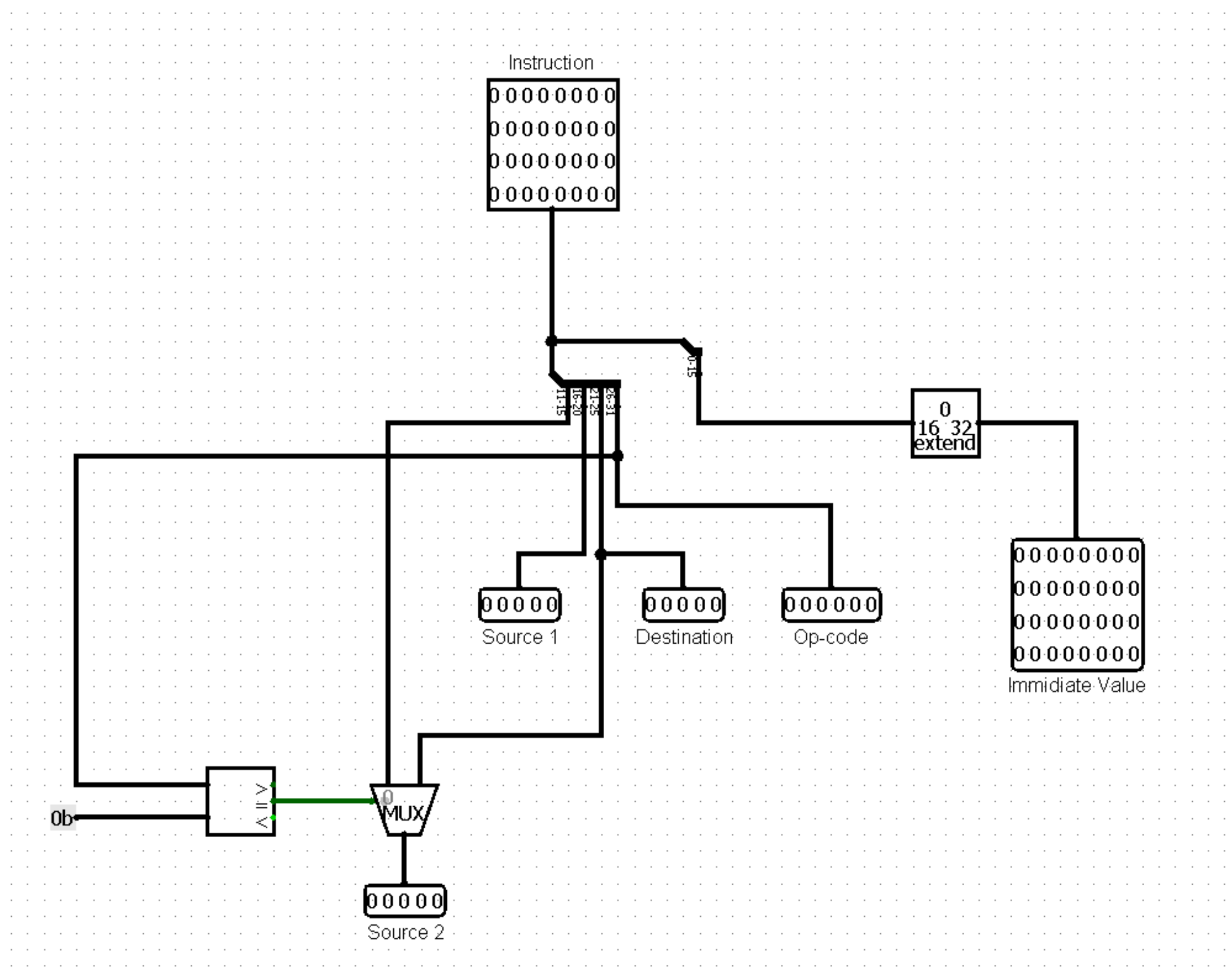
- PC is incremented by 1 so that it gets the address of the next instruction to be executed. Whenever the PC is enabled, the Instruction Register is also to be enabled so that PC gets loaded with the 32 bit instruction located in the memory.

MEMORY UNIT IMPLEMENTATION



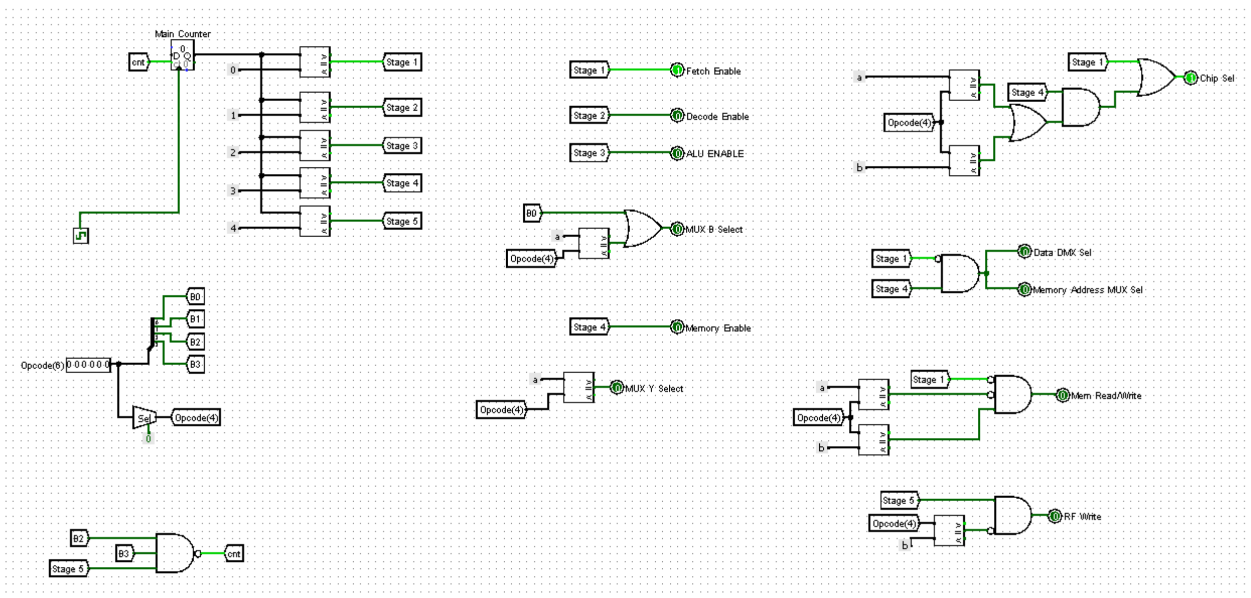
- In RAM, as the maximum address width can be 24 bit, in logisim, we are taking only 24 bit of total 32 bit using the bit selector.
- In RAM, we put the read/write pin for ld, str used for the LOAD/STORE command.
- Chip Sel is responsible for switching the RAM on or off. When its value is 1, memory(RAM) is ready for access and when it is 0, RAM is switched off. All the operations of Chip Sel are controlled by the Control Unit.

INSTRUCTION DECODER IMPLEMENTATION



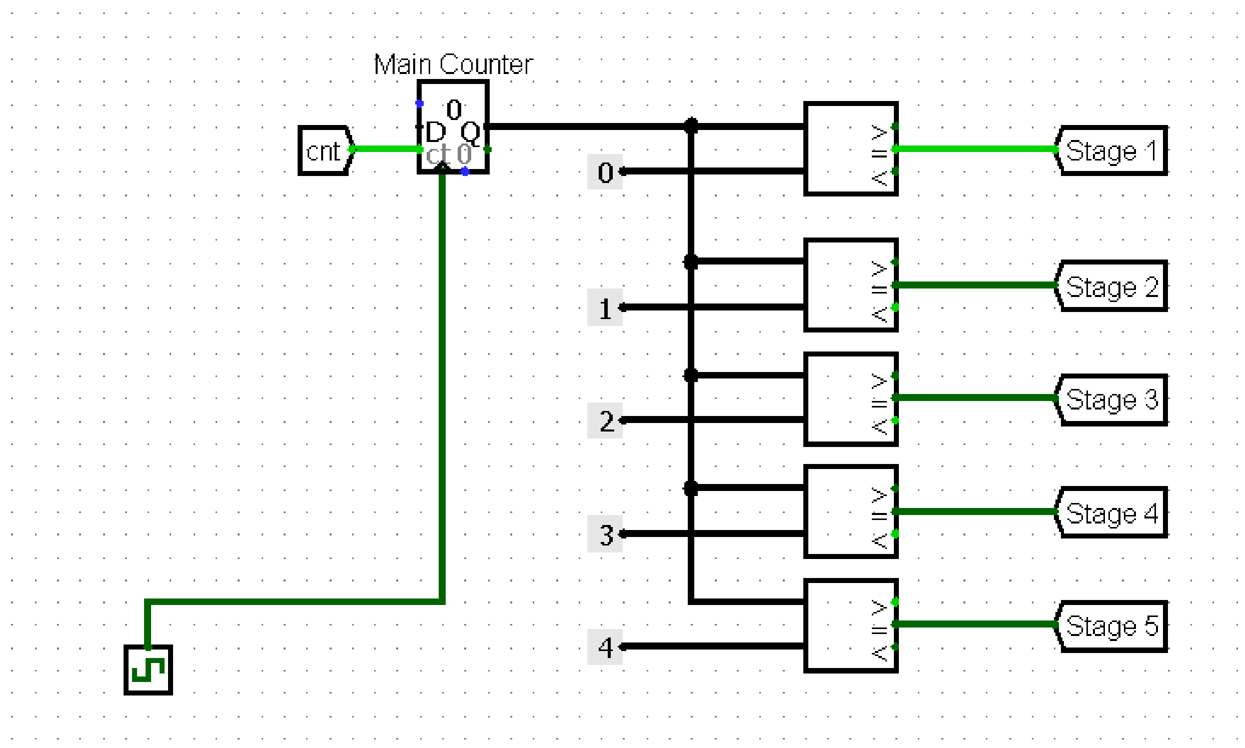
- We get the 32-bit instruction code in instruction input. And then, we separate the instruction into different parts such as op-code(6 bit), destination register(5 bit), source register 1(5 bit), source register 2(5 bit) and immediate value(16 bit).
- When the instruction doesn't need immediate value, it enables source 2, otherwise it enables immediate value. The immediate value is converted from 16 bit to 32 bit using 0 padding. This 0 padding is done by the bit extender as shown in the figure.

<u>CONTROL UNIT IMPLEMENTATION</u>



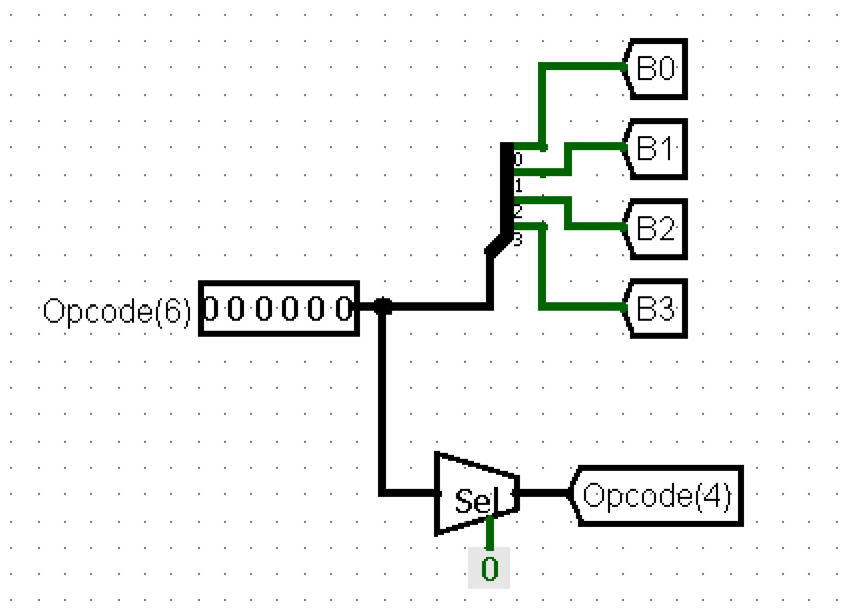
CONTROL UNIT COMPONENTS:

1)



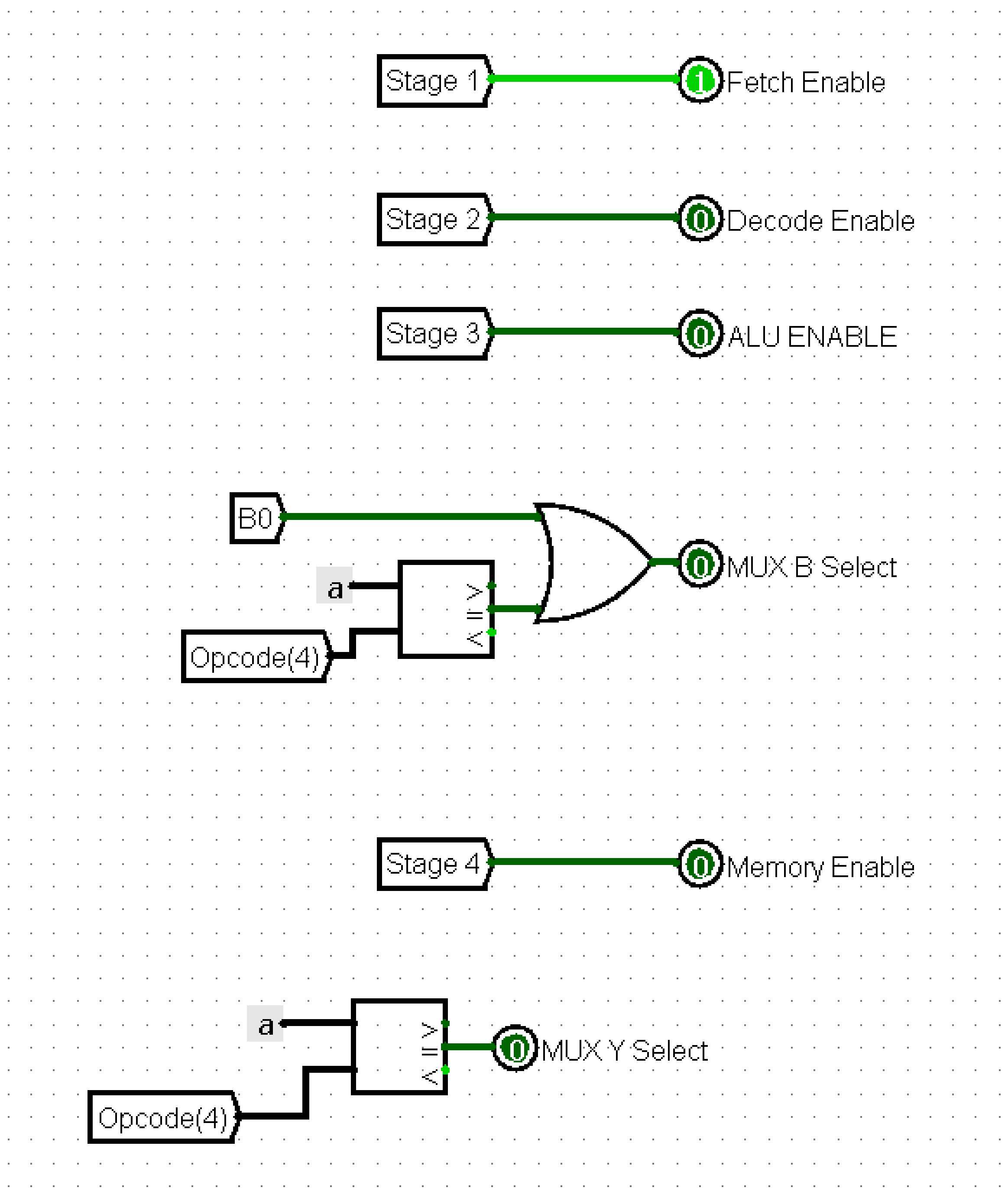
Here, the counter is used to ensure that each line of instruction written passes through all 5 stages in the pipeline. The counter has a wrap after 4 so that after stage 5, it again goes to 0 to start stage 1 for the following instruction line. Comparators are used to activate only one stage at a time.

2)



We have taken op-code of length 6, but not all bits are used to identify the instruction. Only the last 4 digits of the op-code is used. So, we are taking out the last 4 bits of the opcode using a bit-selector and naming it as Opcode(4) in a tunnel to be used at other parts of the Control unit. Also, we need each of the 4 bits of new Opcode(4), so we bifurcated the opcode and named it as B0, B1, B2 and B3,with B3 being the MSB and B0 being the LSB.

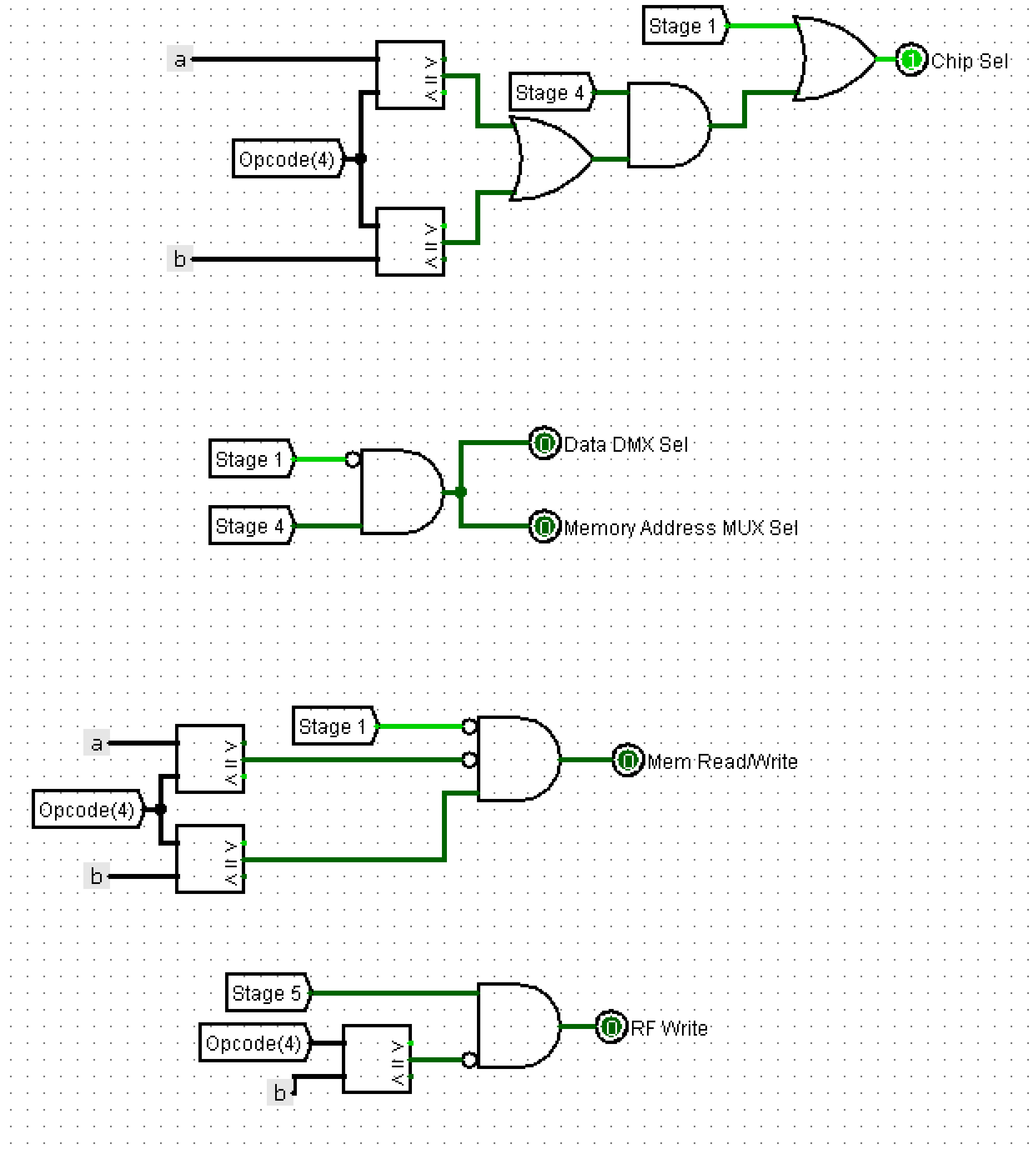
3)



- In stage 1, as Fetch is done, fetch enable is used there.
- In stage 2, as instruction is decoded from the IR, so **Decode Enable** is connected to stage 2.
- In stage 3, ALU computation happens(if required), so **ALU Enable** is linked to stage 3.

- In stage 4, as memory read/write takes place in case of load/store, **Memory Enable** is attached with this stage.
- In commands using immediate, LOAD and STORE, the MUX B turns on. So, we have put all immediate value and STORE in the encoding scheme in such a way that all have least significant bit as 1. In addition to that, we also attached the LOAD to it using a comparator to compare it with the Opcode for LOAD i.e. 1010 (a) as that wasn't covered by the least significant digit thing.
- MUX Y is selected in LOAD command, so we are turning it on using a comparator that compares op-code with 1010(a)

4)



- The Chip Sel is like an enable pin for the ram memory. Thus Chip Sel=1 means memory access is required while Chip Sel=0 means that memory access is not required. Memory access is required in stage 1(for fetching instructions from memory) and in stage 4 for load(opcode =A) and store(opcode=B) operations.

- The Data DMX Sel decides if the memory data fetched is for the load(stage 4) or fetch operation(stage 1). The same logic holds for Memory Address MUX Sel select line.

- The memory read/ write is active in stage 1(fetch instruction) and load(opcode A) and store(opcode B) instruction.
- The RF Write is the parameter which decides the instructions where writeback to one of the registers in the register file is required. So writeback is required in all our operations when stage 5 is active (except store operation). Hence the negation is required in one of the AND gate terminals.