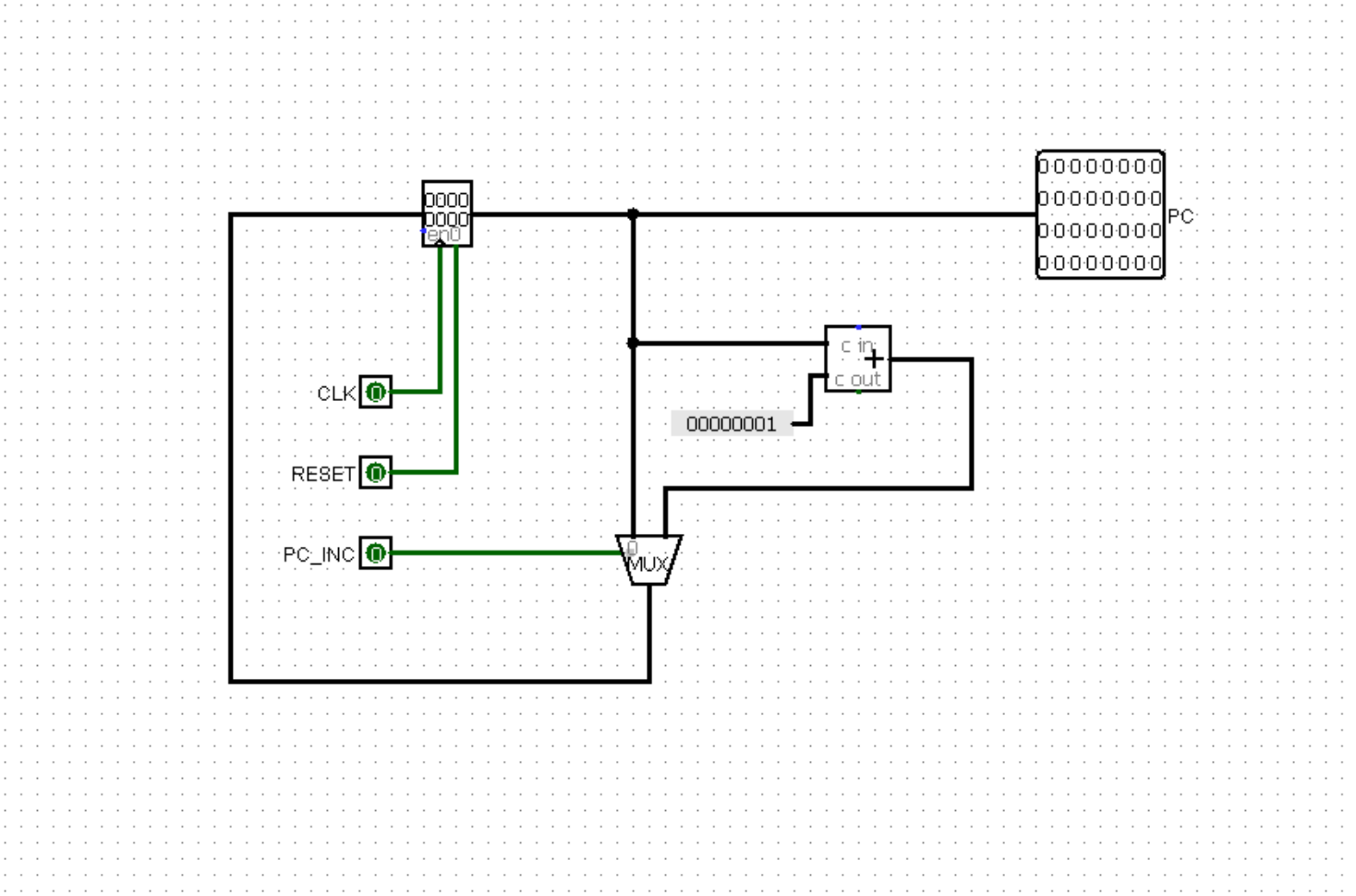
SISD - Report

Five Stage Architecture Principle: -

The whole processor architecture is based on fie architecture principle which has the following stages:

1) Fetch: The content of memory at the address pointed by program counter is fetched into Instruction Register and program counter is incremented as to point to next instruction.
2) Decode: The control unit decodes the Op-code inside the IR (Instruction Register) and based on this value; the control signals are generated. The operands for the instruction are also fetched and stored into Ra and/or Rb in this stage.
3) Execute: Based on the control signal, the ALU (Arithmetic Logic Unit) performs addition, subtraction, logical AND, logical OR, or simply bypasses the value to Rz.
4) Memory: If the instruction is Load/Store, the results of ALU (the effective address of the data) interacts with memory unit to load/store the data else the result is passed to Ry.
5) Writeback: In this stage, if the instruction requires the value to be written back to register file, the value from Ry is written back to register file.

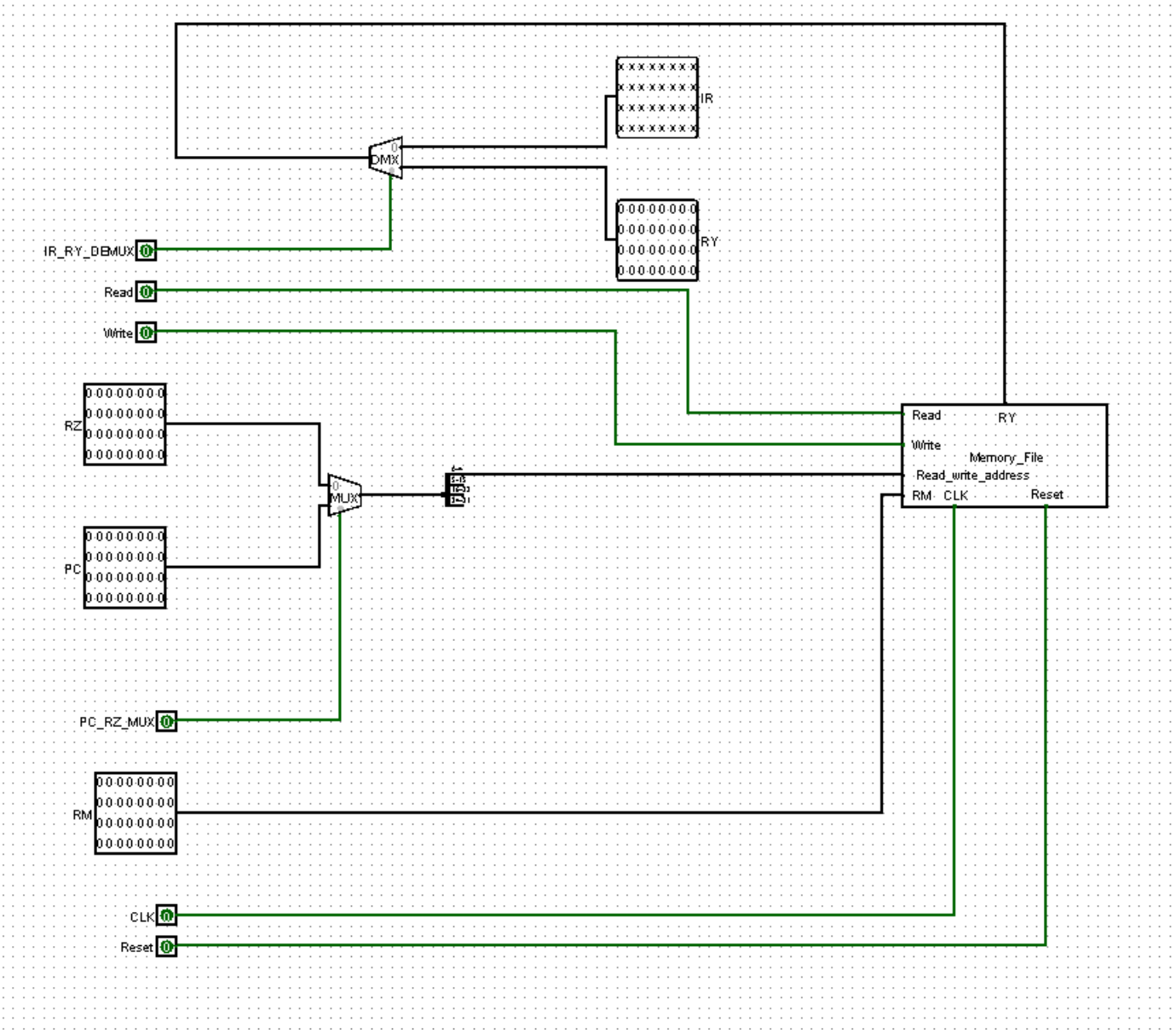
PROGRAM COUNTER:



It points to the instruction currently being executed in the memory.
The PC_Inc value decides whether PC should be increased or remain the same. When PC_Inc = 1, PC increased by 1 (Fetch stage), otherwise remains the same. The Reset pin asynchronously clears the data in the program counter.

MEMORY:

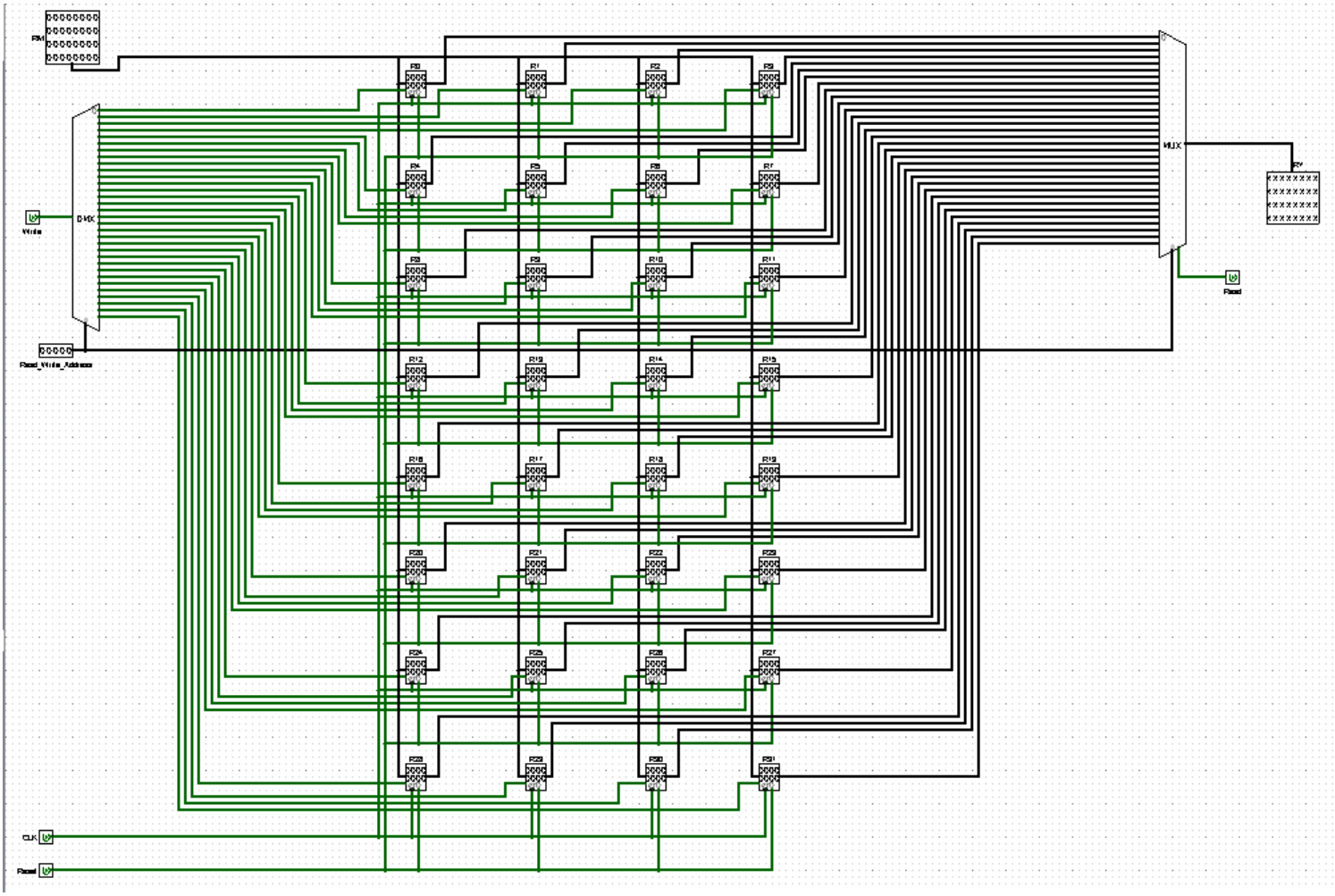
Memory Management Unit:



It interfaces processor with the memory.

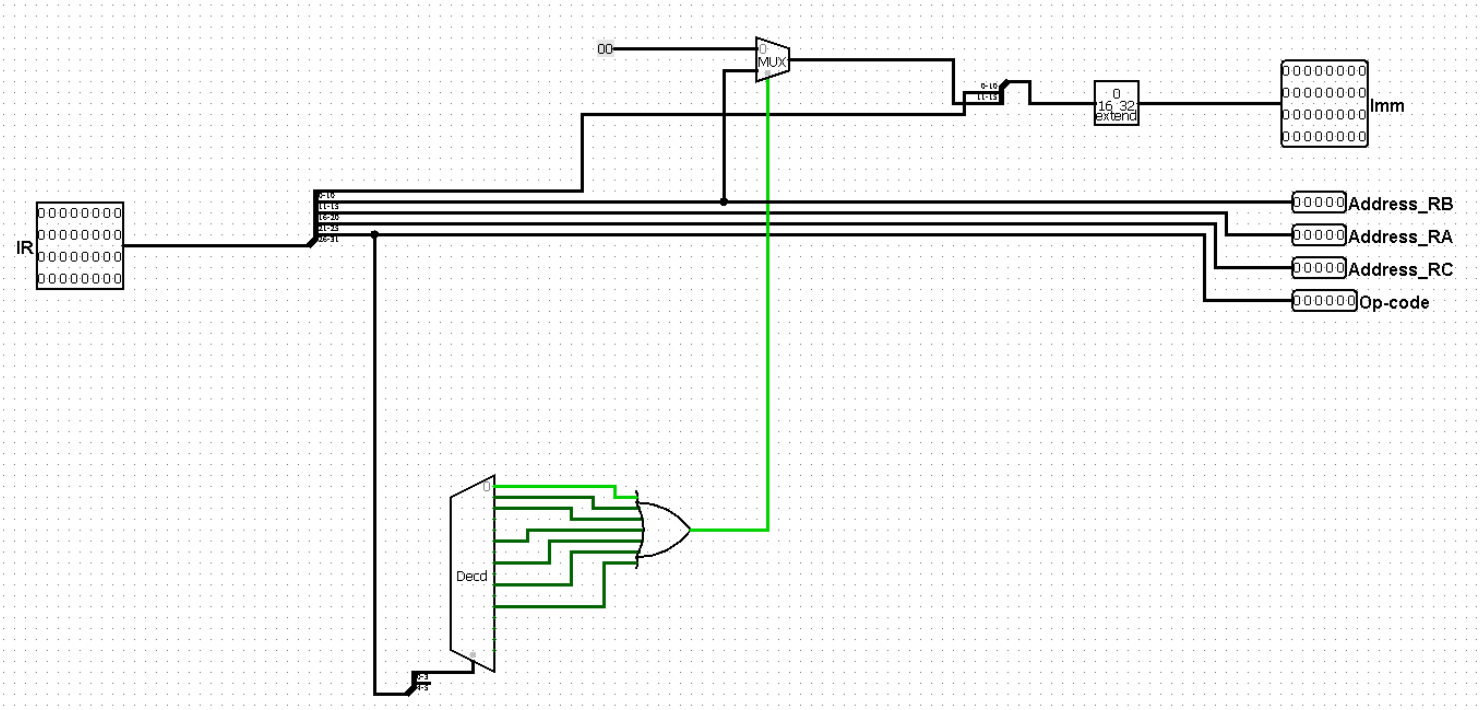
The Data from the Memory File can go to either IR (in Fetch stage) or RY Register (in memory stage of instruction, say, load) depending on the control signal IR_RY_Demux. The Data from RM can go in Memory File (in memory stage of instruction, say, store) only. The Address bits for accessing memory can come from either RZ (if the address is computed/bypassed by ALU) or PC (in case of fetching the instructions from the memory). As the memory has 8 bit address line, LSB 8 bits are selected from PC or RZ based on control signal PC_RZ_MUX.

Memory File:



- It has 32 registers for storing the data in memory (R0 – R31).
- The writing to Memory-File is selected based on write control signal, when Write = 1 which enables writing of RM to one of the 32 registers in Register-File based on the control signal Read_Write_Address.
- The data is read through RY based on the address provided in Read_Write_Address, when Read = 1.
- The Reset pin asynchronously clears the data in all the registers.

BIT SPLITTER:



It separates different types of bits from the instruction register and feeds them to different components.

The encoding scheme is shown below:

(31-26) Op-Code

(25-21) DESTINATION

(20-16) SOURCE1

(15-11) SOURCE2

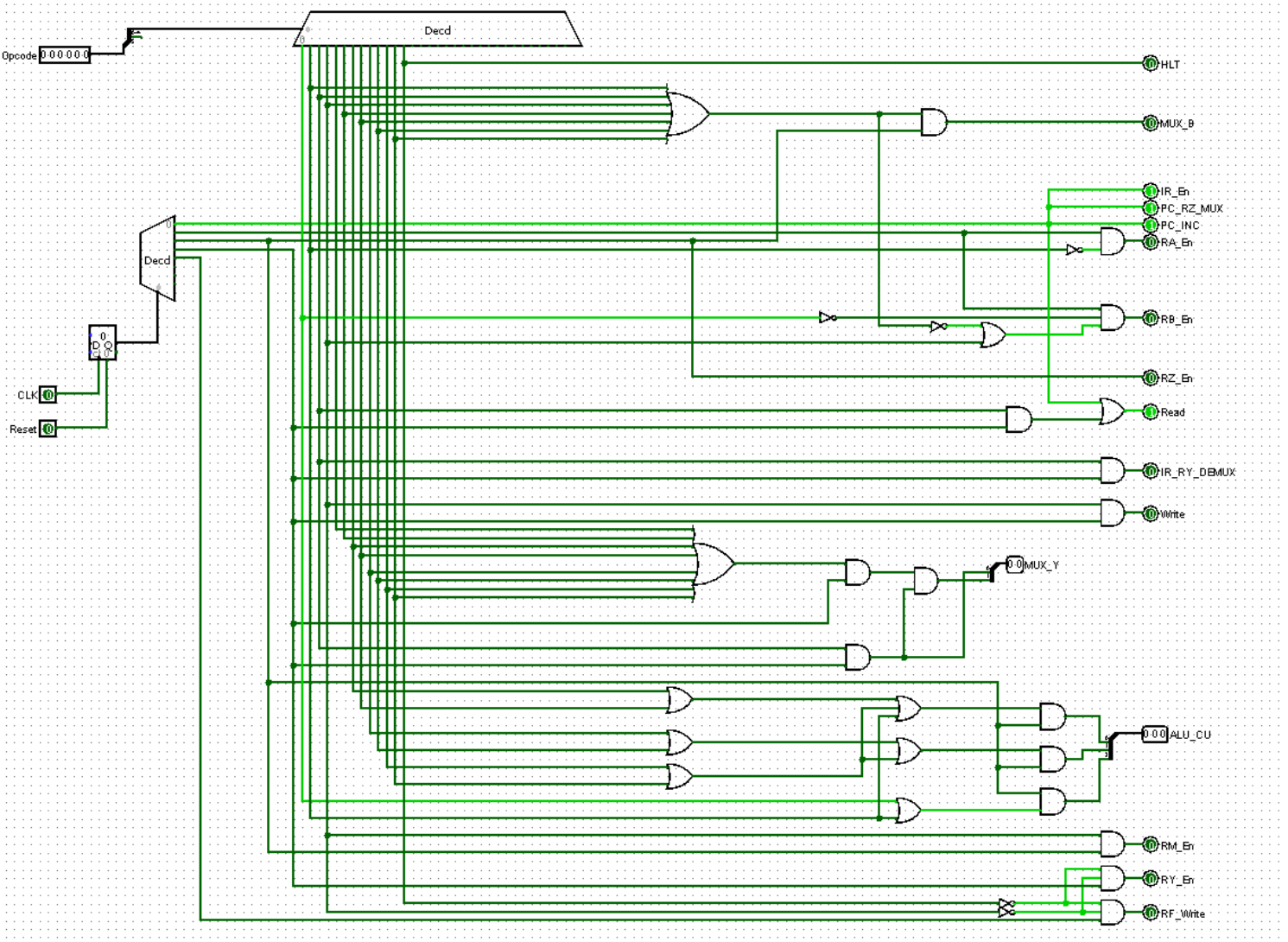
(15-0) Immediate

When we use all the three address RA, RB and RC, there are only 11 bits left for Immediate value. But when RB is not used the 5 bits used to store address of RB from the first 5 bits of Immediate.

Due to this difference in bit sizes, we use a multi-plexer to decide the first 5 bits of Immediate value from the set of all zeros or address RB.

The select line for the above MUX is calculated from the opcode corresponding to different operations.

CONTROL UNIT:



The opcode thus obtain from the bit splitter is further used to decide the operation. This along with the current stage is used to decide different control signal generated by Control-Unit.

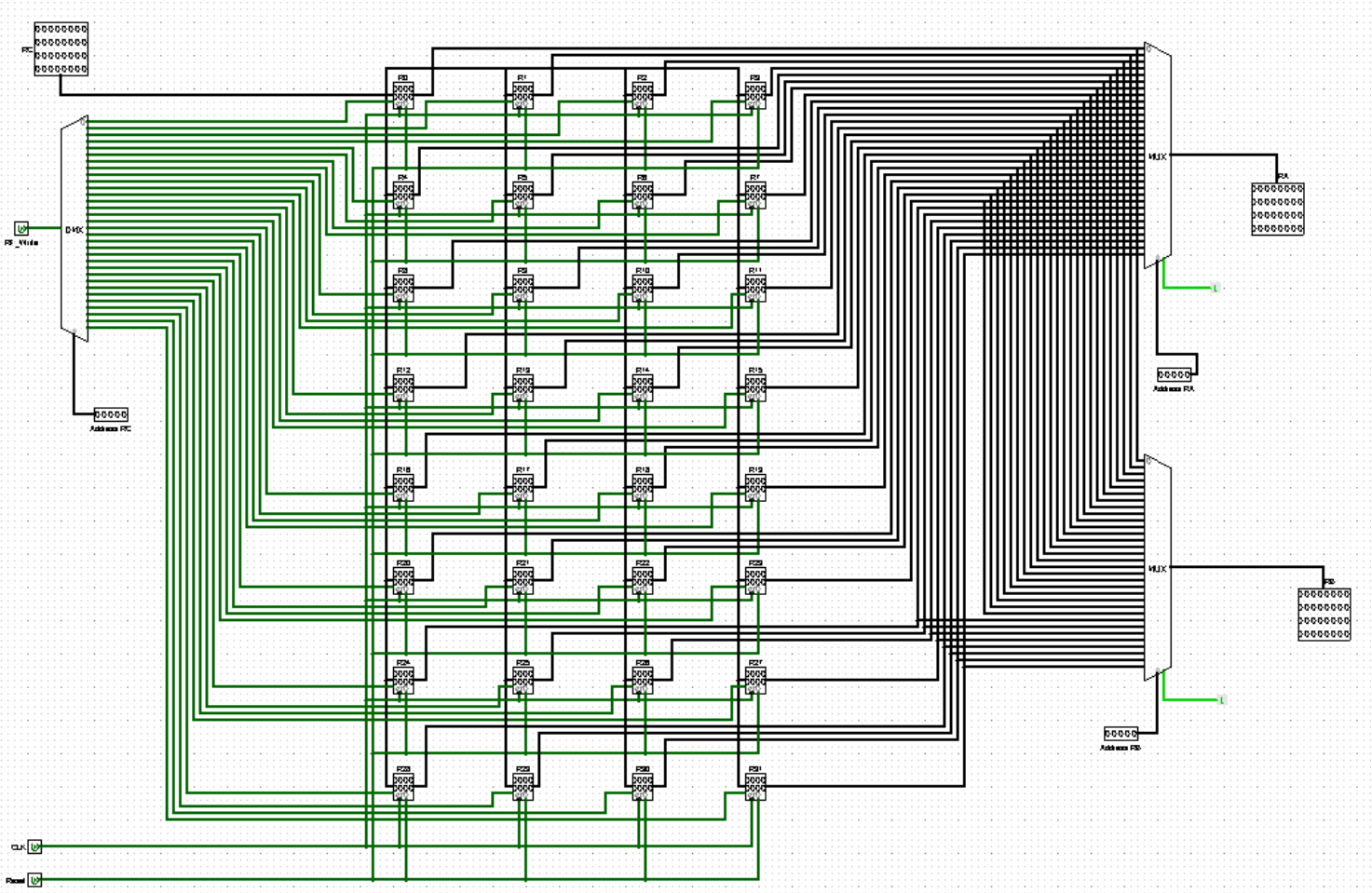
To generate different stages 1 through 5, a counter is used with maximum value set to 5 in series with a decoder.

Control unit generates various control signals that include: -

1. **IR_En**: This control signal is generated at first stage to store the instruction in IR.
2. **IR_RY_DEMUX**: This signal used as select line of DEMUX that take value from Memory-File and corresponds to RY or IR accordingly.
3. **PC_RZ_MUX**: This control signal used as select line in MUX that select the address value from first 5 bit of current PC (at stage 1) or RZ (at stage 4).

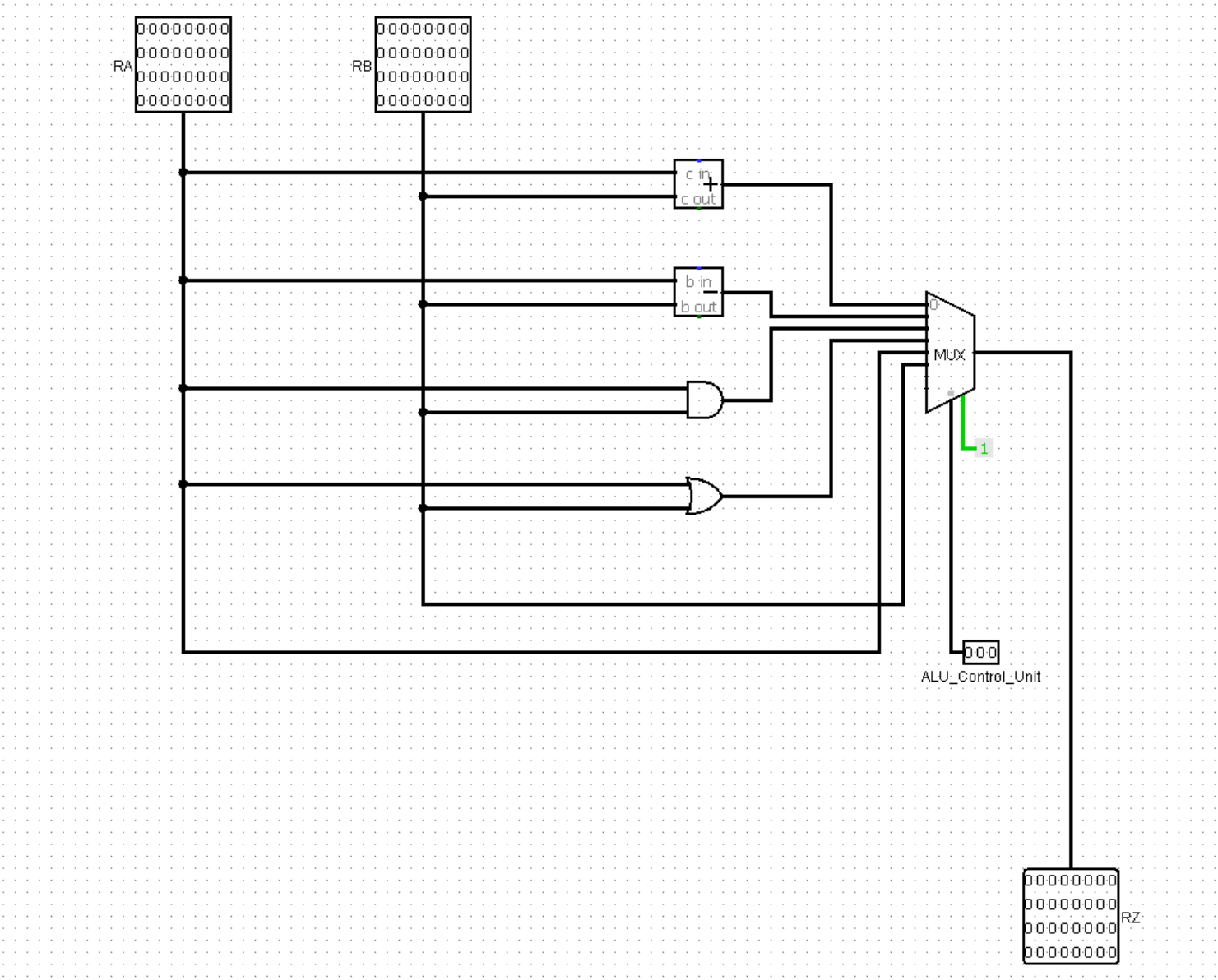
4. **PC_Inc:** This control signal is used as a select line for incrementing the value of current PC. This too is used in stage 1 itself.
5. **READ**: This signal is used to perform read operation from memory-file.
6. **RA_En**: This signal is used to store value from Register-file to RA register (at stage 2).
7. **RB_En**: This signal is used to store value from Register-file to RB register (at stage 2).
8. **RM_En**: This signal is used to store value from RB to RM register (at stage 3). The value of RM further passed to Memory-File if required.
9. **MUX_B**: This signal is used as select line of MUX that take input RB and Immediate and select accordingly.
10. **ALU_CU**: This control signal is used to perform various operation that is done in ALU.
11. **RZ_En**: This signal is used to activate register RZ in order to store the value calculated by the ALU.
12. **Write**: This signal is used to perform write operation in Memory-File.
13. **MUX_Y**: This signal is used as select line in MUX operating between RZ, Read (Memory-File), Return Address (Return operation not considered in this project).
14. **RY_En**: This signal is activated when we have to store value in register RY.
15. **RF_Write**: This signal is used when we have to write back to Register-File.
16. **HLT**: This signal is used to reset value from all the registers that are used in the whole circuit.

REGISTER-FILE:



- It has 32 registers for storing the data (R0 – R31).
- The writing to Register-File is selected based on write control signal, when write = 1 which enables writing of RC to one of the 32 registers in Register-File based on the control signal Address_RC.
- The data is read through A and B based on the address provided in Address_RA and Address_RB, when write = 0.
- The Reset pin asynchronously clears the data in all the registers.

ARITHEMETIC LOGIC UNIT (ALU):



- It is a combinational circuit that supports the following operations: ADD, SUBTRACT, AND, OR, MOVE (without immediate), MOVE (with immediate) on RA and RB and produces result as RZ.
- The operation being performed is selected using the control signal: ALU_Control_unit.

SEQUENCE OF OPERATIONS IN FIVE STAGE ARCHITECTURE:

STAGE 1:

- Instruction register is updated with 32-bit instruction code from memory location pointed by the program counter.
- After the read operation from memory to IR is completed the value in PC is incremented by 1 in order to points towards the next instruction.

STAGE 2:

- The instruction code inside the Instruction register is decoded/split into sections comprising of the opcode, Address_RC, Address_RA, Address_RB and the Immediate value (Imm value is extended from 16-bit to 32-bit for further operation).
- The register file receives the address of RA and RB and then updates the values in register RA and RB outside the Register-File.

STAGE 3:

- As per the operation, the value of RB and Immediate are selected through a MUX and passed to ALU for further calculation.
- The ALU receives proper control signals from the Control-Unit and processes the input(s) as per the opcode using the available combinational circuits.
- The result is then stored into the register RZ.
- Additionally, the value in RM is updated with the value of RB for memory operation if needed.

STAGE 4:

- Then, for store operations such as LOAD/STORE, the value in RZ is passed to the address location in memory.
- For LOAD, a memory read control signal is generated by the CU that activates the memory read MUX so that data can be copied from memory and passed to the RY register via MUX_RY under appropriate control signals.
- For STORE operations, the value to be stored in memory is passed through register RM. The value computed in RZ is passed to the memory write address. In this situation where there is a data and address location for a STORE operation, the CU generates a memory write signal that updates the value stored in the memory register pointed to by the memory address

- If no memory operation is performed in stage 4, the value in RZ is passed to the RY register under the correct MUX control signals.

STAGE 5:

- The value thus produced is then sent back to the Register File with write back signal from CU.