KUZEY KIBRIS KAMPÜSÜ
ODTÜ METU
NORTHERN CYPRUS CAMPUS

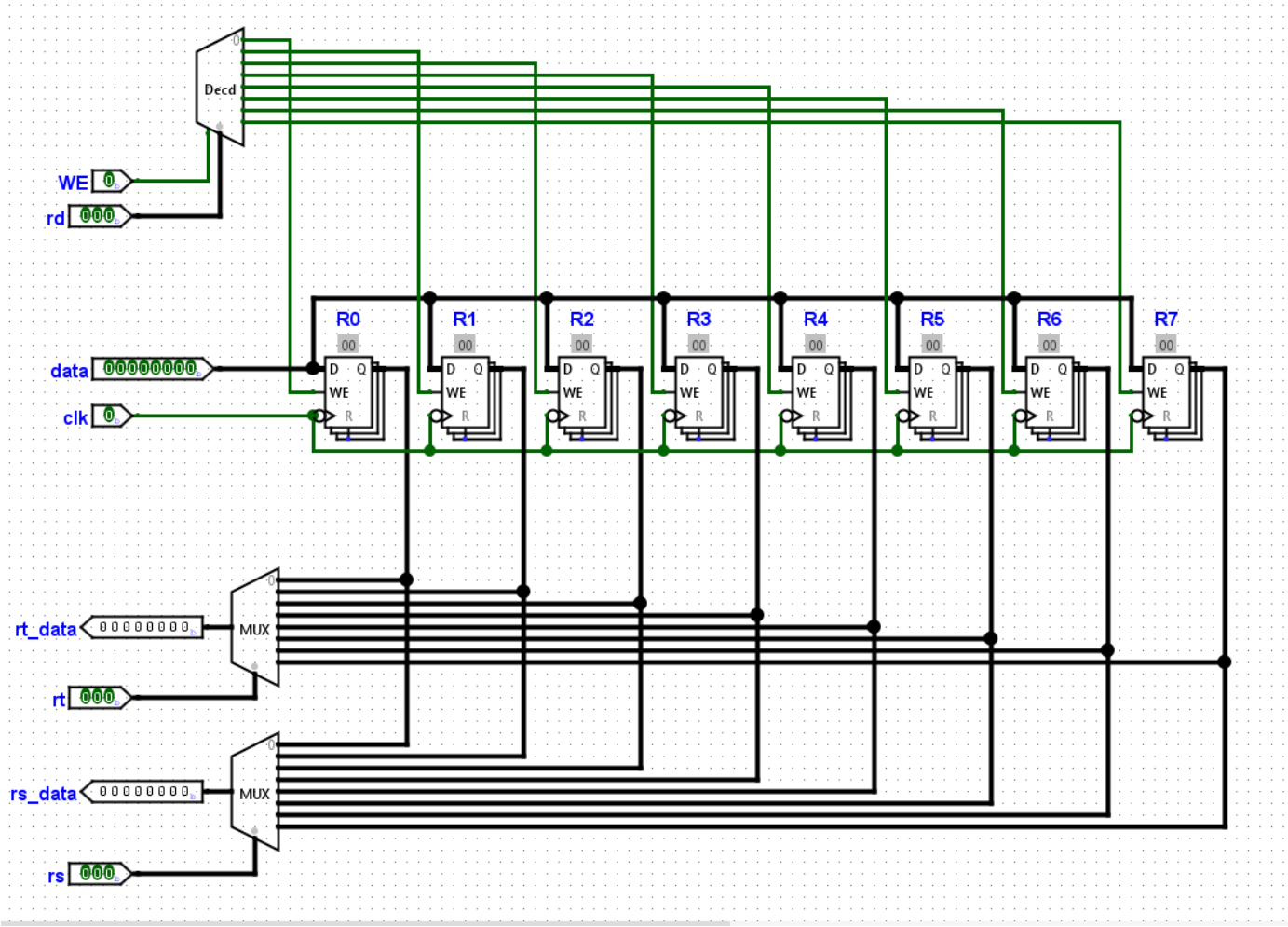
# CNG 331 Project 3 Report

**Name:** Muhammad Somaan
**ID:** 2528404

# Part 1)

# Register File:

### Schematic:



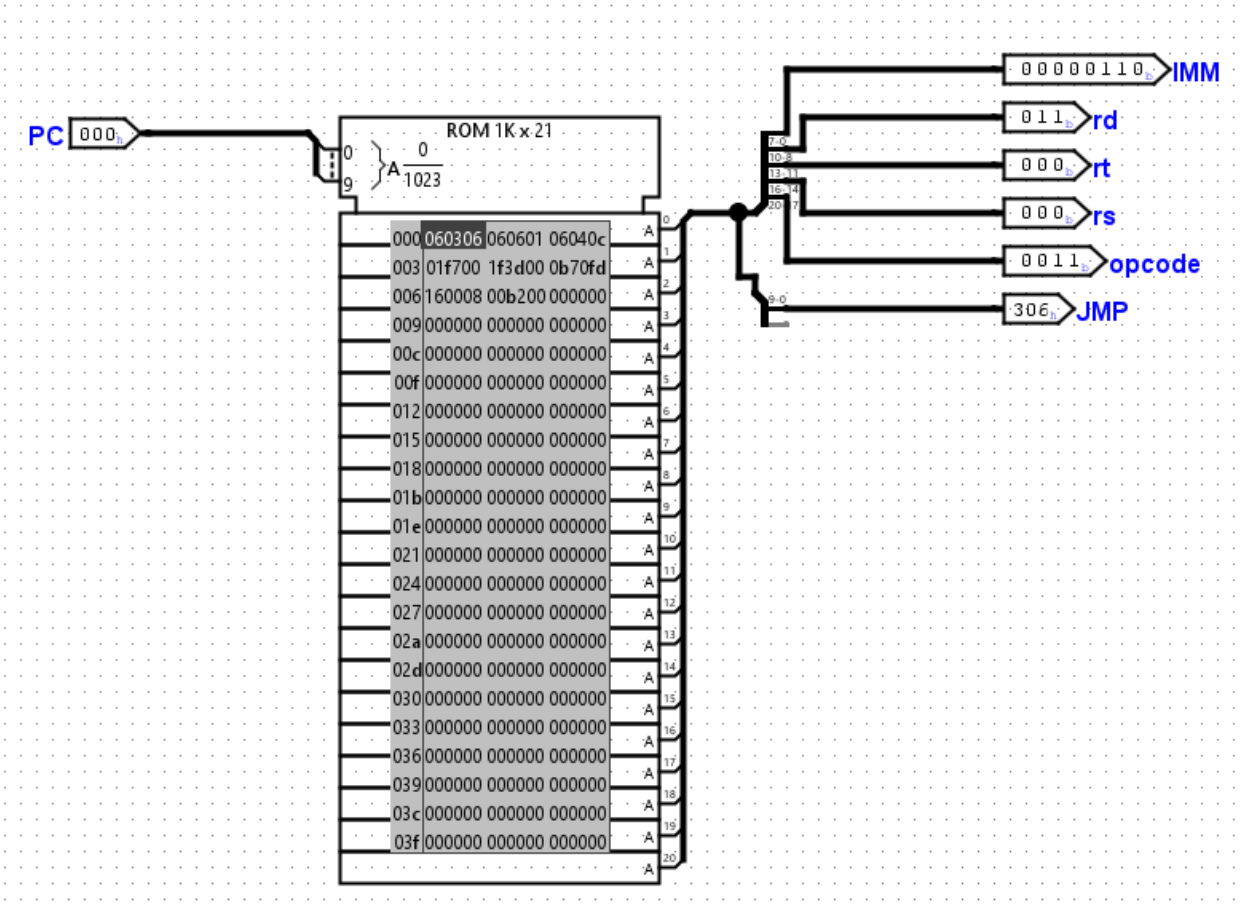

Comments: Register R0 to R7 are individual registers, all share the same clock and rt_data and rs_data is the output depending on the select rt and rs. To write data to registers, we provide WE input as 1 and rd select to specify the register to store the input data in.

Do we need enable for mux? No we do not since we want mux to work all the time.

# Part 2)

# Instruction Memory:

## Schematic:



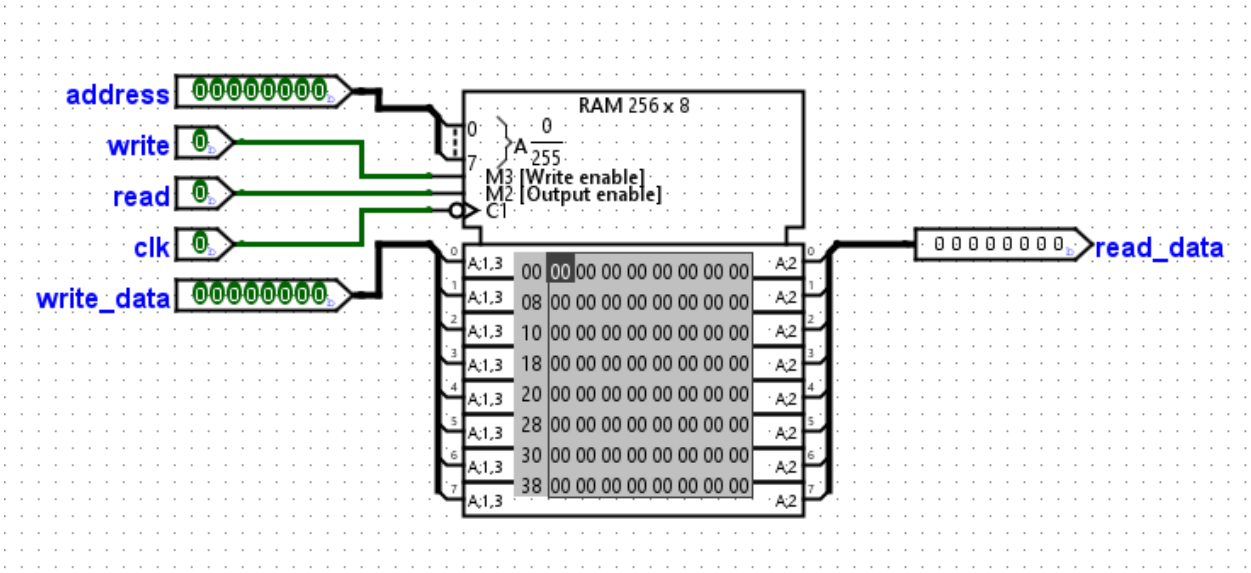
## ROM Properties:

| ROM "Instruction_Memory" | |
|---|---|
| FPGA supported: | Supported |
| Address Bit Width | 10 |
| Data Bit Width | 21 |
| Line size | Single |
| Allow misaligned? | No |
| Contents | (click to edit) |
| Label | Instruction_Memory |
| Label Font | SansSerif Bold 16 |
| Label Visible | No |
| Appearance | Logisim-Evolution |

Comments: ROM stores a 21 bit data, since our instruction length is 21 bit. The address bit width is 10 bits, allowing for 1024 instructions to be stored. ROM takes PC input and fetches that instruction and outputs the 21 bit data, into 5 parts. Opcode, Rs, Rt, Rd and Immediate value IMM. JMP is the immediate value for jump instruction.

# Data Memory:

## Schematic:




## RAM Properties:

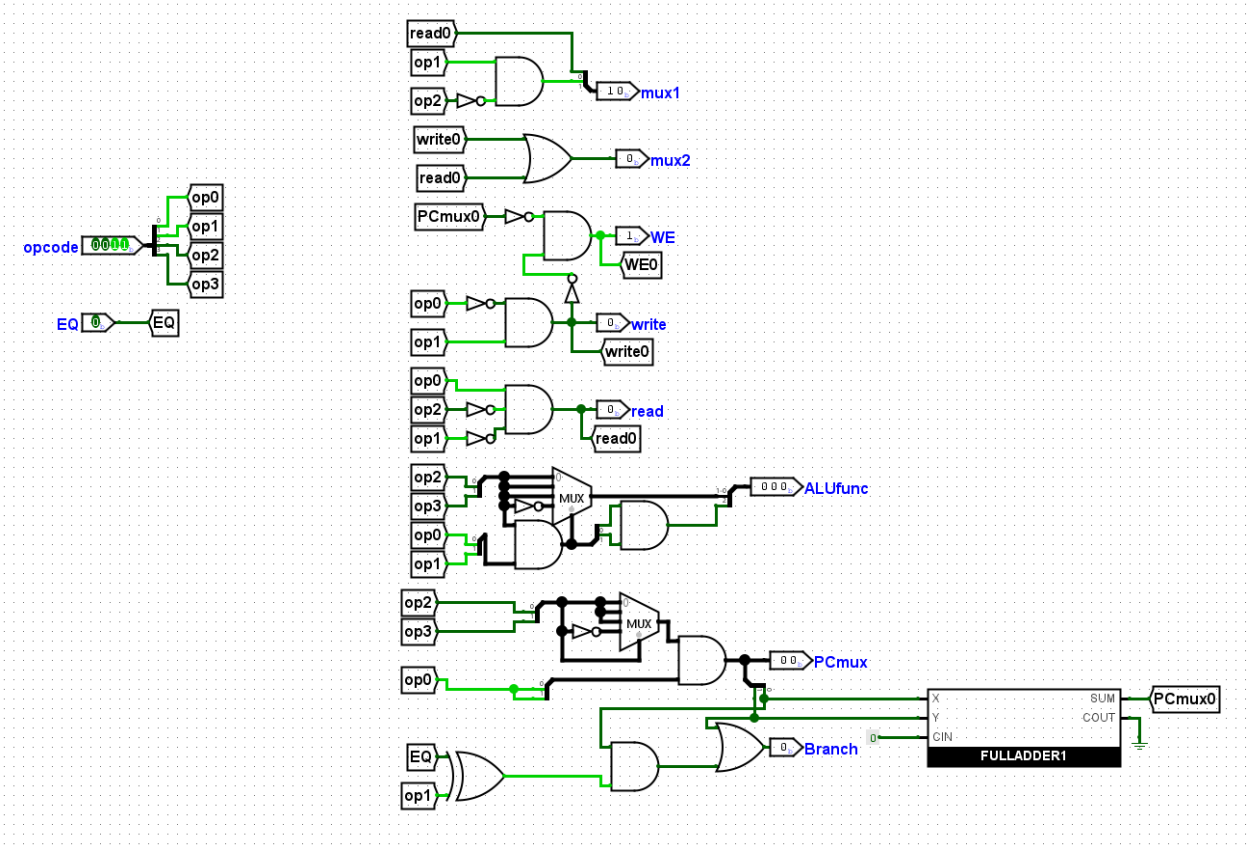
| RAM "My_Ram" | |
|---|---|
| FPGA supported: | Not supported |
| Address Bit Width | 8 |
| Data Bit Width | 8 |
| Enables: | Use byte enables |
| Ram type | volatile |
| Use clear pin | No |
| Trigger | Falling Edge |
| Asynchronous read: | Yes |
| Data bus implementation | Separate data bus for read an... |
| Label | My_Ram |
| Label Font | SansSerif Bold 16 |
| Label Visible | No |
| Appearance | Logisim-Evolution |

Comments:  RAM stores 8 bit data with 8 bit address. My data memory takes address and data as input and then takes write or read flags as inputs to determine if data is to be read from the RAM or it should be stored. I provide clock clk as input to keep all clocks in CPU the same. Read_data outputs the data in the address provided if it is a read operation.
Does the RAM store your values as big endian or little endian? It stores it in hexadecimal big-endian.

# CPU)

# Control Unit:

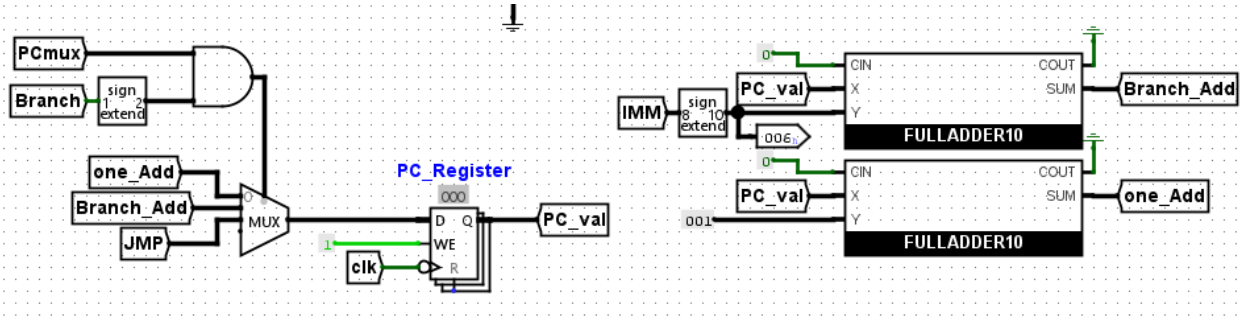
### Schematic:




Comments: opcode is taken as input to decide upon different outputs. Mux1 and Mux2 output is the select for mux1 and mux2 in cpu, WE is write enable for register file and then read and write output are flags for data memory, and ALUfunc is the output for deciding the alu function. PCMux is the output for mux 3, and Branch is the output for selecting the branch. All the outputs are calculated from the truth table given in the question paper.

# Program Counter:

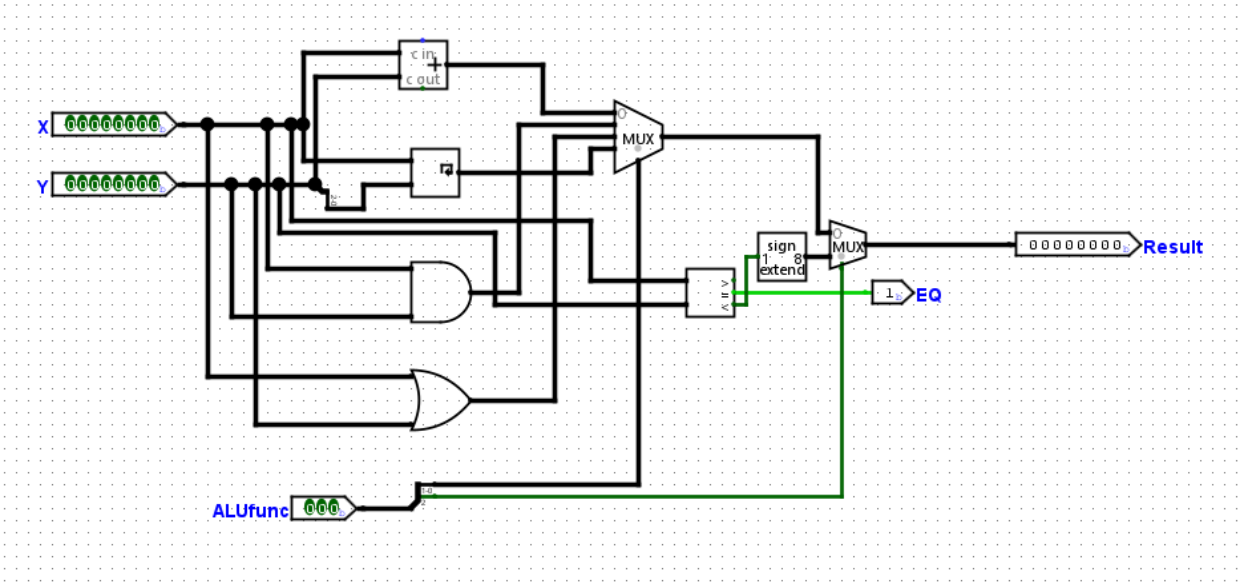
## Schematic:




Comments: PC is implemented using a register, and Branch_Add and One_add are the addition value after adding the branch immediate or 1 to PC's current value.

# ALU:

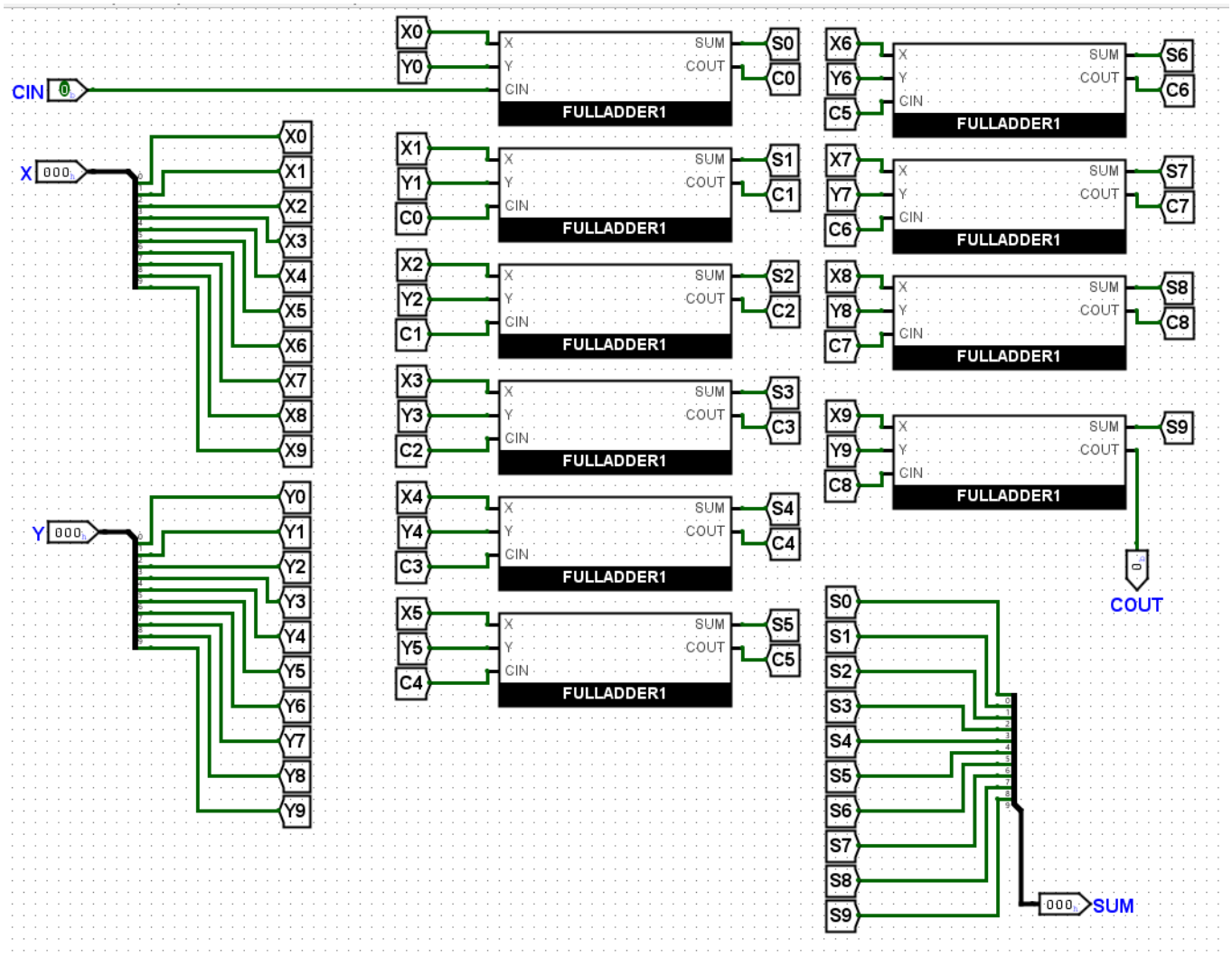
## Schematic:




Comments: EQ is added as output, to determine if X and Y are equal.

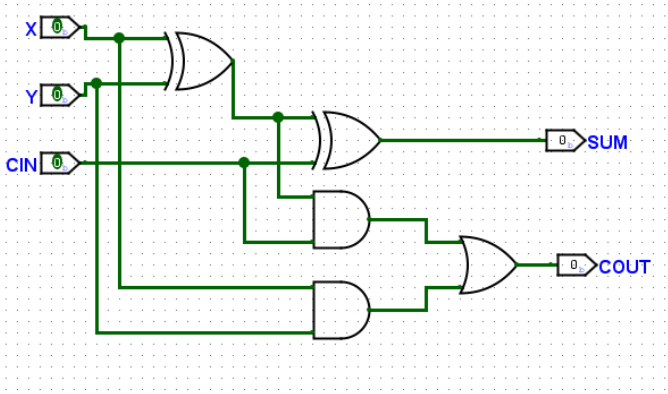
# FULLADDER10:

## Schematic:



# FULLADDER1:

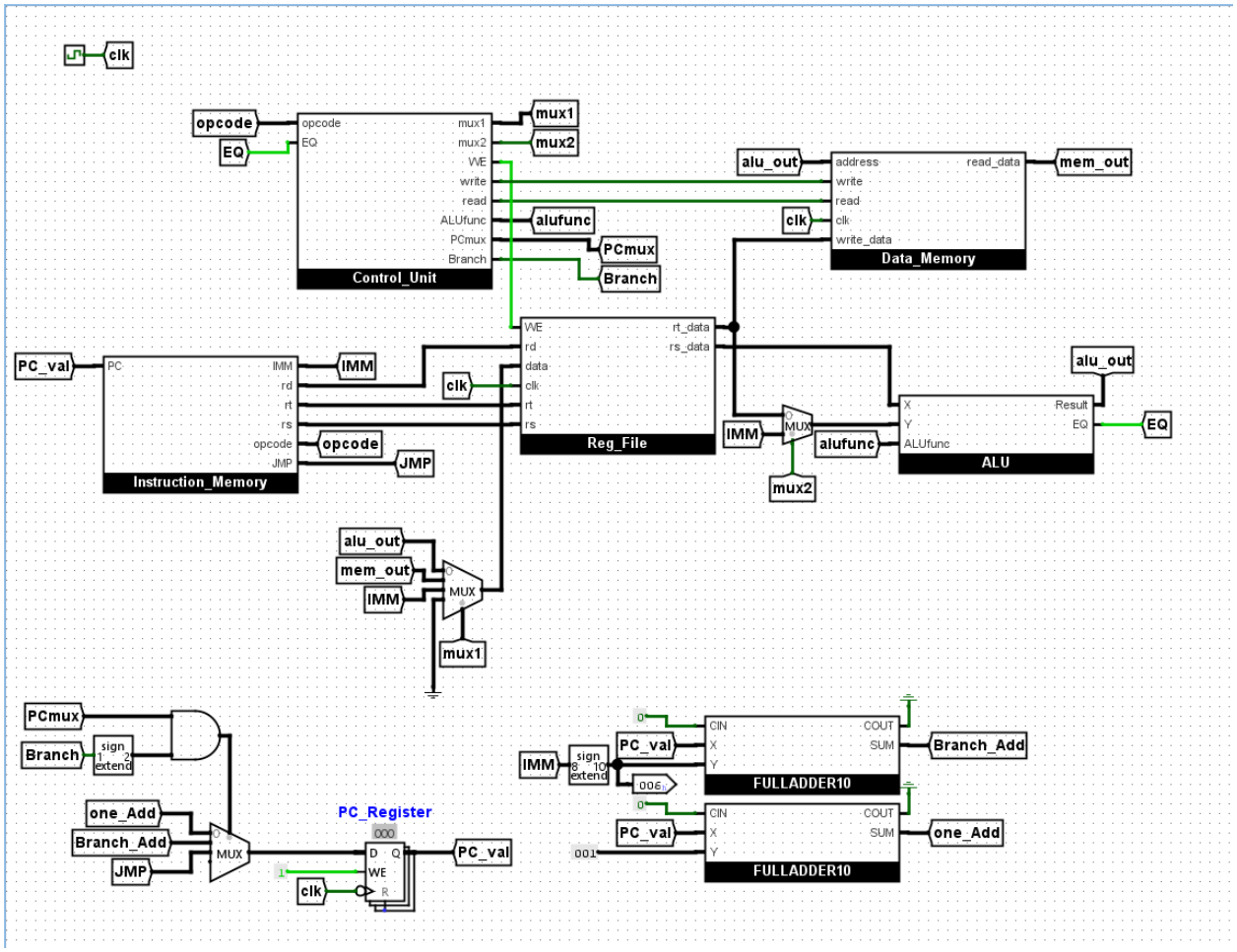
### Schematic:




Comments: My program counter takes clk clock as input for that all clocks in CPU are same. Reset input is used to reset the pc. Program Counter is a register which starts from 0 and then for every clock cycle it is incremented by 1. For addition a 10 bit full adder is used. After incrementing it outputs the value to PC_val.

# CPU:

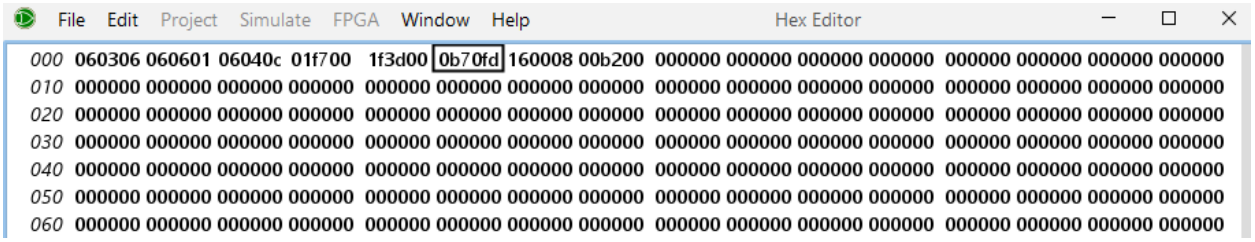
## Schematic:



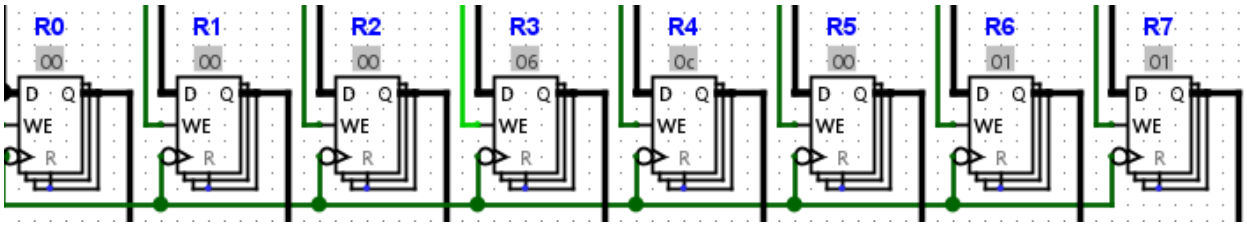

Comments: CPU is designed according to the schematic provided.

# TEST)

# Hex File:




# Register Contents:




Comments: First 6 is loaded in R3, followed by storing 1 in R6 and then storing 12 in R4. Afterwards R6 is added to R7 and which gives 1. Then R4 and R7 are compared and checked if R4 is lower than R7, since it is not, R5 is set to 0. Now R6 and R5 are compared, since they are not equal there is no jump to L1. now the program jumps to the end, and skips the addition instruction in the end since R5 and R6 are not equal.