CSE 3015 Project Report

Rıdvan Gülcü – 150117508

Ali Berat Çetin – 150117822

About The Project

In this project we are expected to design a CPU which supports a set of instructions, 16 registers with 18 bit data width. Processor will have 18 bit address width also. Instructions are AND,OR, ADD, LD, ST, ANDI, ORI, ADDI, XOR, XORI, JUMP, BEQ, BGT, BLT, BGE, BLE. CPU consists of 7 main components. These are **Register File, Instruction Memory, Data memory, Control unit, Alu, Branch Operator.** Instruction memory is a read-only for instruction data set. Data memory is read-write for load and store operations. Memory has 10 bit address width. Alu is responsible for XOR,OR,ADD,AND and immediate types of these instructions. Branch operator is a part that we decode the branch opcode then decide by NZP values to generate an output whether condition Is true. We also wrote an assembler to test and debug our CPU design.

Instructions

ADD Destination,Source1,Source2 adds source1 and source2 and store into destination.

ADDI Destination,Source,Immediate adds source1 with an immediate value and store into destination.

XOR Desination,Source,Immediate xor operation between source and immediate value and store into destination. XORI,OR,ORI,AND,ANDI is the same. Just gate differences.

JUMP Address sets the Program Counter(Pc-relative) with given address value.

LD Destination, Address loads data from Data Memory with given address to Destination register.

ST Source, Address stores from Source register into Data Memory with given address.

BEQ Operand1,Operand2,Address compares the equality of op1 and op2, then set the Program Counter.

BLT Operand1,Operand2, Address compares op1 and op2. If op1 is less than op2 sets the Program Counter.

BGT Operand1,Operand2, Address compares op1 and op2. If op1 is greater than op2 sets the Program Counter.

BLE Operand1,Operand2, Address compares op1 and op2. If op1 is less than or equal to op2, sets the Program Counter.

BGE Operand1,Operand2, Address compares op1 and op2. If op1 is greater than or equal to op2, sets the Program Counter.

Assembler

Assembler is the part of the project where converts assembly code into binary. There is a map to op-codes from assembly instructions. We wrote that assembler according to our ISA table. We used java programming language for fast development and convenience of the array operations.

Instruction Set Architecture Table

Rıdvan Gülcü 150117508 - Ali Berat Çetin 150117822

	[17:14](opcode)	[13:10]	[9:6]	[5:2]	[1:0]
AND	0010	DR	SR1	SR2	00
ANDI	0011	DR	SR1	IMM6	
OR	0100	DR	SR1	SR2	00
ORI	0101	DR	SR1	IMM6	
ADD	0110	DR	SR1	SR2	00
ADDI	0111	DR	SR1	IMM6	
XOR	1000	DR	SR1	SR2	00
XORI	1001	DR	SR1	IMM6	
JUMP	1010	ADDRESS14			
LD	1011	DR	ADDRESS10		
ST	1101	SR	ADDRESS10		

BEQ	1110	OP1	OP2	0	1	0	ADDRESS3
BGT	1110	OP1	OP2	0	0	1	ADDRESS3
BLT	1110	OP1	OP2	1	0	0	ADDRESS3
BGE	1110	OP1	OP2	0	1	1	ADDRESS3
BLE	1110	OP1	OP2	1	1	0	ADDRESS3

Figure 1. ISA Table

Since we have 16 instructions, we allocated 4 bits for opcodes. We didn't use all combinations of 4 bit because we need to group branch opcodes due to design restrictions of control unit. For immediate instructions, immediate limits [-32,+31]. Since we have 16 registers we allocated 4 bits for register selectors.

While designing control unit, we put a 16x1 decoder for control unit states. Because of this we grouped the branch instructions. There are two fetch steps and 16 instructions. They didn't fit 16x1 decoder. To make a change on that would effect our assembler. We are decoding branch operations based on NZP values after opcode fetch. Address limit of the branch instructions is between [7,0].

Register File

Register file has input load, clock, indata, write and output read1 and read2. Read1 and Read2 are selectors for the registers which we want to read. Write(WR) is selector for register which we want to write data in it. When Load is 1, indata is stored to Destination register specified by write. When load is 0, registers preserver their values.

There are 16 registers have 18 bit data-width.

1- InData : 18 bit data input

2-Clock : shared clock for all system

3-Write : 4 bit input to select register which to be loaded.

4-Read1 & Read2 : 4 bit input to select registers which to be readed.

5-Load : Signal to enable write operation on registers.

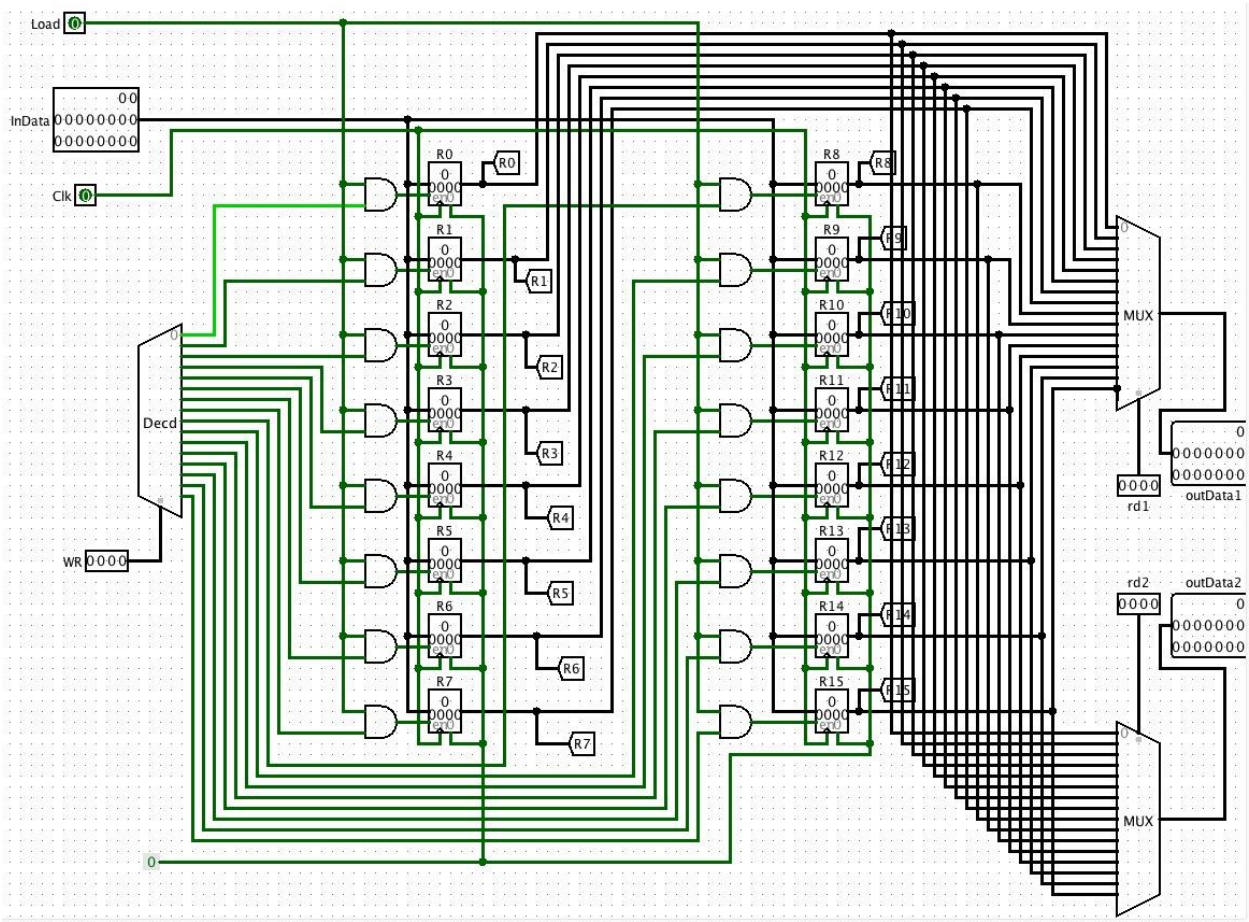

Figure 2. Register File

Control Unit

Control unit decodes opcodes and generates signals for data paths.

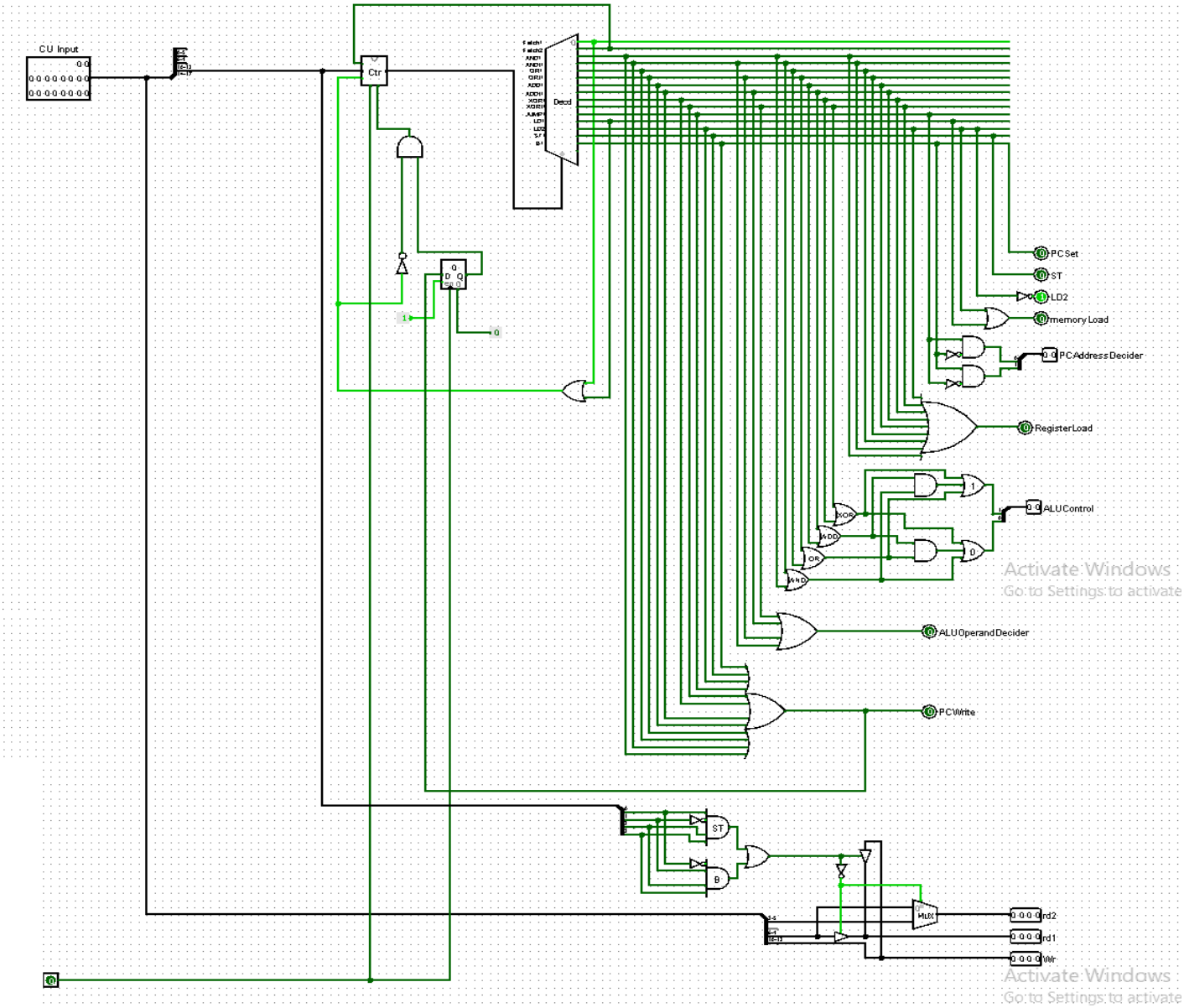

Figure 3. Control Unit

We implemented a FSM to control states of the Control Unit. If operation is LD MEM LOAD and REGISTER LOAD signal will be 1. If operation if ST MEM WRITE signal will be 1. LD operation takes two clock cycles. So there are LD1 and LD2 in FSM. First And second outputs of the decoder are fetch1 fetch2. If operation belongs to ALU, for immediate instructions immediate signal will be 1 otherwise 0.

Alu Control is 2 bit signal. Because there are 4 types of operation in ALU. XOR,AND,OR,ADD. ALU instructions and LD instructions enable REG WRITE signal. Because there is Destination register in these instructions. Comparing operation(branch) and jump operation Control Unit generates signals to select address bits of the jump and branch operations. PC write is set just before fetch1 state to be ready for new instruction after fetch2 state.

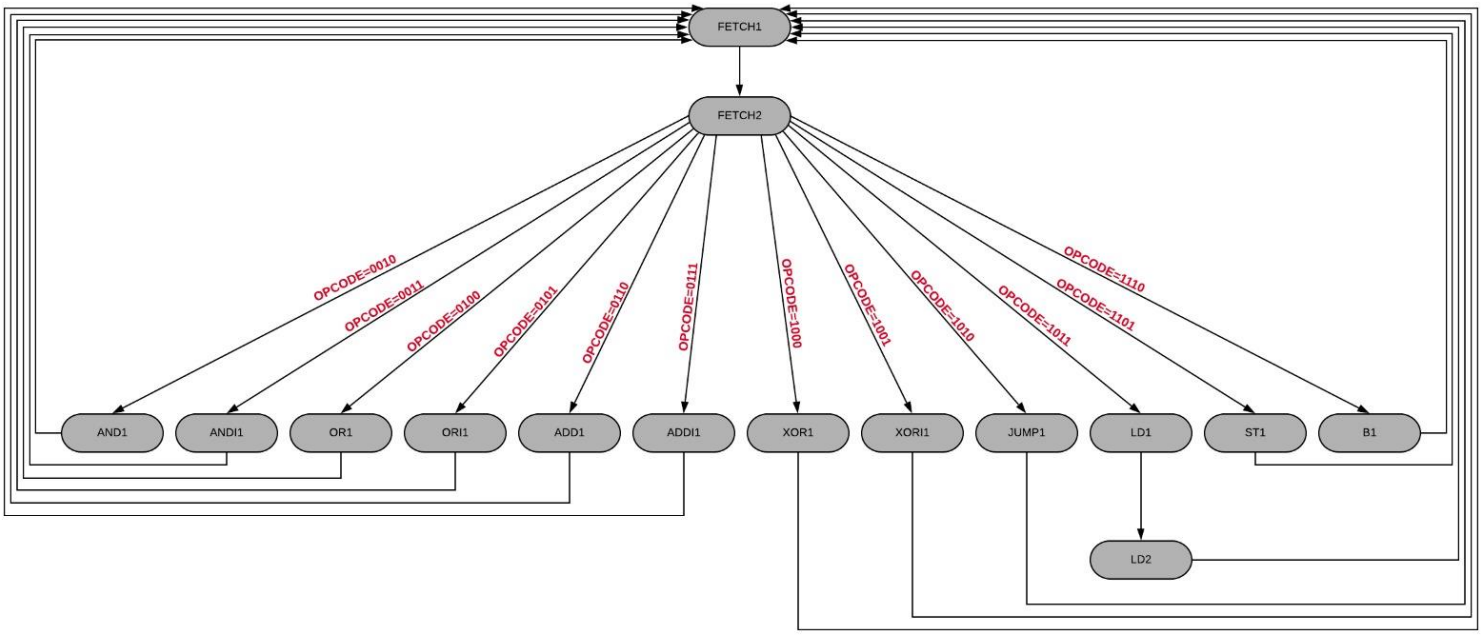

Figure 4. Finite State Machine Diagram

First, we designed a Finite State Machine to make easier to design Control Unit circuit. If an instruction is finished, state goes back to fetch1.

Inputs								Outputs													
Opcode				Current State				ST	LD2	memoryLoad	PCAddressDecider		RegisterLoad	ALUControl		ALUOperandDecider	PCWrite	Next State			
[3]	[2]	[1]	[0]	[3]	[2]	[1]	[0]	[0]	[0]	[0]	[1]	[0]	[0]	[1]	[0]	[0]	[0]	[3]	[2]	[1]	[0]
X	X	X	X	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0
0	0	1	1					0	1	0	0	0	0	0	0	0	0	0	0	1	1
0	1	0	0					0	1	0	0	0	0	0	0	0	0	0	1	0	0
0	1	0	1					0	1	0	0	0	0	0	0	0	0	0	1	0	1
0	1	1	0					0	1	0	0	0	0	0	0	0	0	0	1	1	0
0	1	1	1					0	1	0	0	0	0	0	0	0	0	0	1	1	1
1	0	0	0					0	1	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	1					0	1	0	0	0	0	0	0	0	0	1	0	0	1
1	0	1	0					0	1	0	0	0	0	0	0	0	0	1	0	1	0
1	0	1	1					0	1	0	0	0	0	0	0	0	0	1	0	1	1
1	1	0	0					0	1	0	0	0	0	0	0	0	0	1	1	0	0
1	1	0	1					0	1	0	0	0	0	0	0	0	0	1	1	0	1
X	X	X	X	0	0	1	0	0	1	0	0	0	1	0	1	0	1	0	0	0	0
X	X	X	X	0	0	1	1	0	1	0	0	0	1	0	1	1	1	0	0	0	0
X	X	X	X	0	1	0	0	0	1	0	0	0	1	1	0	0	1	0	0	0	0
X	X	X	X	0	1	0	1	0	1	0	0	0	1	1	0	1	1	0	0	0	0
X	X	X	X	0	1	1	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0
X	X	X	X	0	1	1	1	0	1	0	0	0	1	0	0	1	1	0	0	0	0
X	X	X	X	1	0	0	0	0	1	0	0	0	1	1	1	0	1	0	0	0	0
X	X	X	X	1	0	0	1	0	1	0	0	0	1	1	1	1	1	0	0	0	0
X	X	X	X	1	0	1	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0
X	X	X	X	1	0	1	1	0	1	1	0	0	0	0	0	0	0	1	1	0	0
X	X	X	X	1	1	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0
X	X	X	X	1	1	0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0
X	X	X	X	1	1	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0

Figure 5. FSM Truth Table

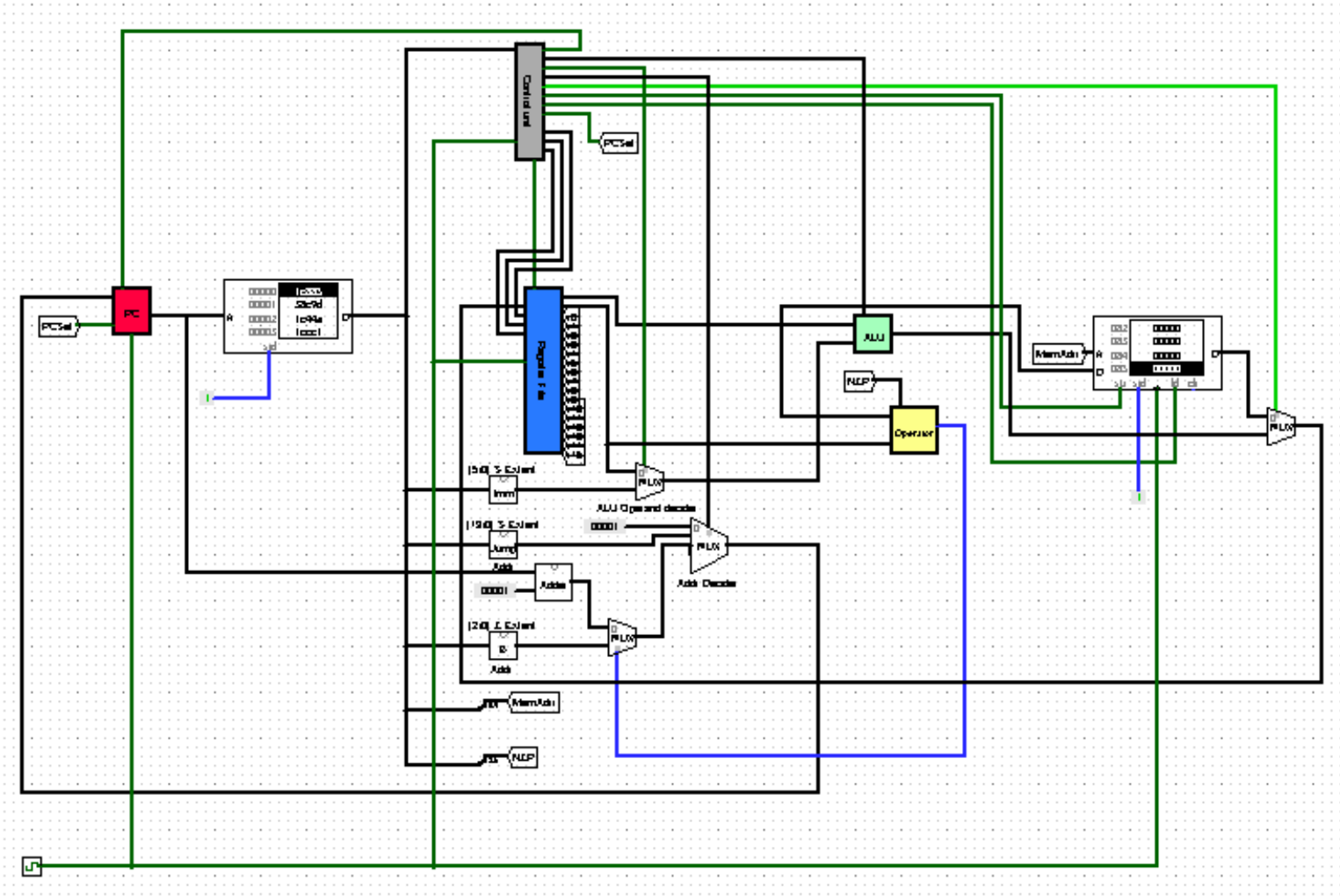

Figure 6. Main

There are several components are used our main CPU design. There are signals come from Control Unit but we also need component to extract information from the instruction memory. For jump instruction, we have a sign extender [13:0] bits. Because jump address could be negative value to set back the Program Counter. Branch Operator is unit that has three input. First is operand1 second is operand2 and third one is NZP value. It computes and outputs whether condition is true or false. According to the output of Branch Operator, PC address is set or not. ALU Operand decider is the part that waits a signal from Control Unit whether the instruction is immediate or register type.

LD and ST instructions use data memory and registers. LD loads data to register from memory, ST stores data to data memory from register. For both, we need a Memory Address that comes with instruction. Memory waits two signals from control unit for ST and LD instruction.

For ST instruction Control Unit signal sets the STORE signal of data memory, for LD instruction Control Unit signal sets the LOAD signal of data memory. There is a mux on the out of the data memory, this

mux decides source of the data to be written to register file. If the instruction is an ALU operation, then source should be from ALU, if the instruction is LD the source should be from data memory.

Instruction[5:0] sign extend for immediate values.

Instruction [13:0] sign extend for jump address.

Instruction [2:0] zero extend for branch address.

Instruction [5:3] for NZP values.

Instruction [10:0] for LD and ST addresses.

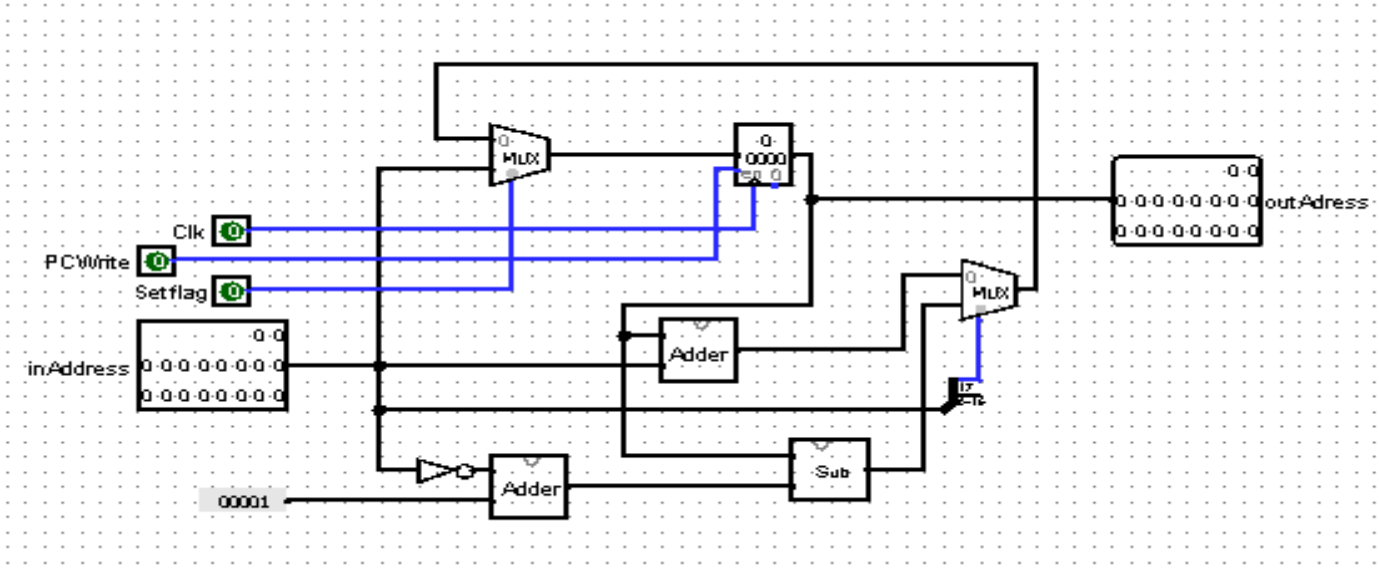

Figure 7. Program Counter

Program counter is responsible for tracing the instructions. It has own logic to increment, jump and set. Set flag is for branch operation to set program counter a specific address. If setflag true, inaddress is assigned to the register. If setflag is false, register value is incremented or decremented by inaddress value.

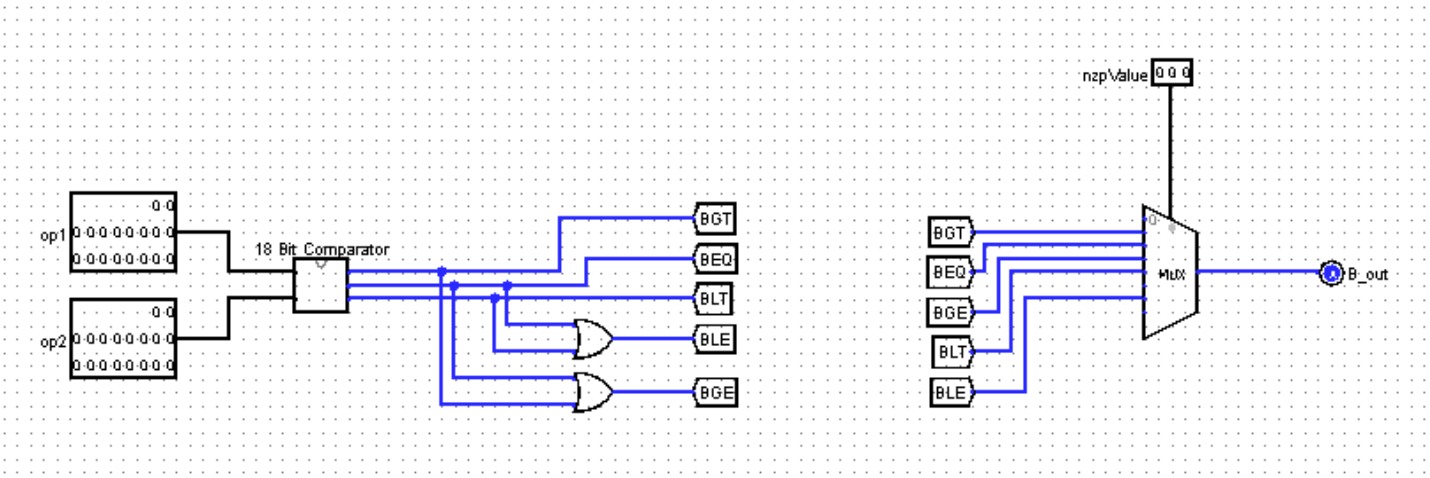

Figure 8. Branch Operator

Branch operator has 3 input. Operand1, operand2 and nzp value. According to the nzp value, one of the computed values of 6 combination of Branch instruction is passing through mux.