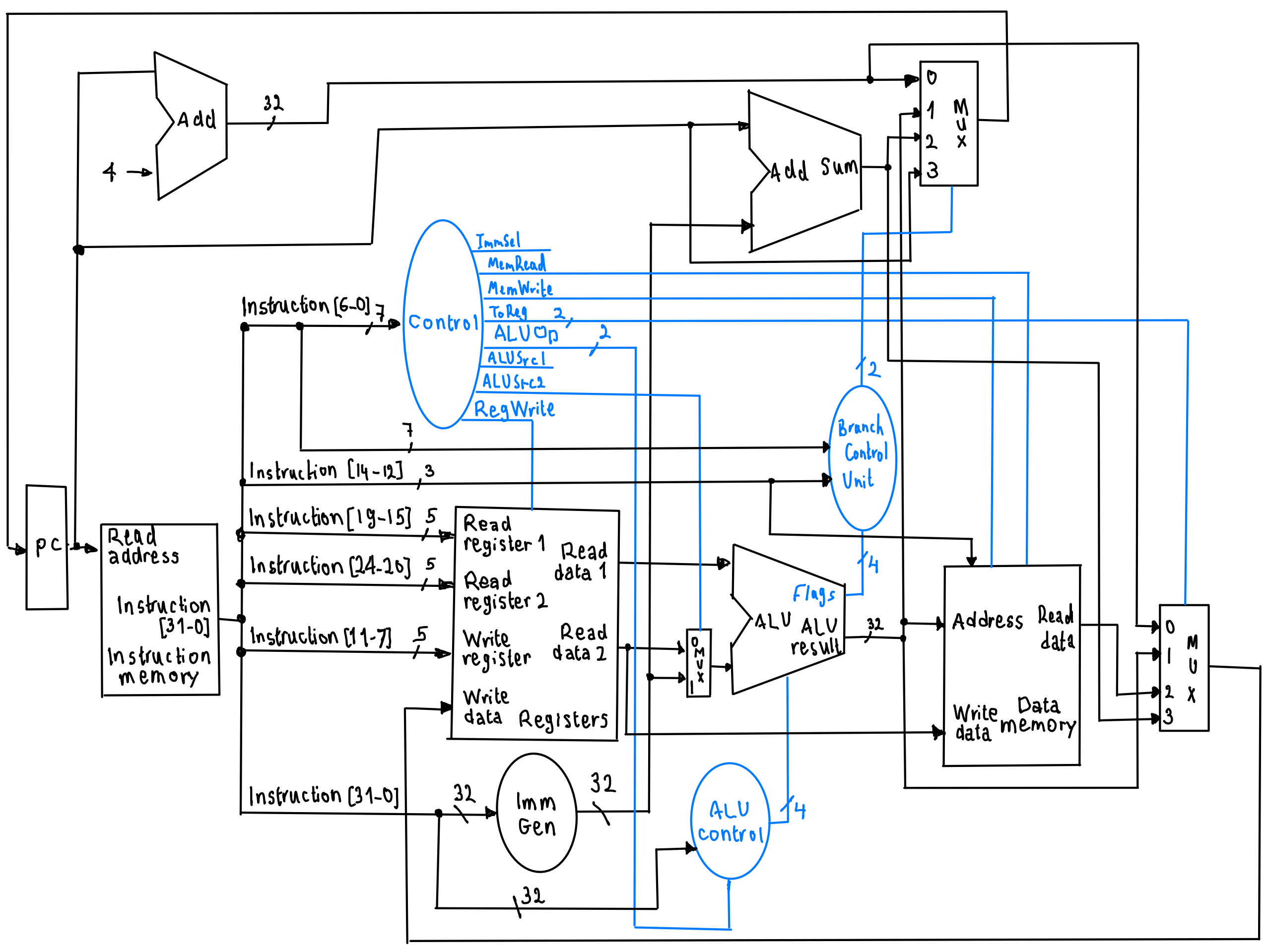
MS2: Single-cycle Datapath for all RV-32I Instructions

Sherif Hisham Gabr 900183120
Mina Ashraf Gamil 900182973

Report

- Final Datapath:



- Design Decisions:

- Initially, we intended to remove the PC target adder and let the ALU compute the target address, by adding a MUX that chooses between rs1 or PC as 1st input to ALU. In that case, there will be one less input to the MUX that decides the write data of the destination register. Another benefit it would have is that the JAL, JALR AUIPC, and Branch instructions would all be computed from the ALU, in that case there will be one less input to the MUX that decides the next PC value. However, we reverted back to the design using the PC target adder, with some additional modifications, because we realized that the ALU is responsible for outputting the needed flags by the branch instructions. So, the ALU is not able to compute the target address and output the correct flags at the same time.

- We added a Branch Control Unit that decides the next PC value. The inputs to this unit are the instruction opcode, instruction funct3, and the flags coming from the ALU. If the instruction is JAL or JALR, the PC will jump to the appropriate address irrespective of the ALU flags. If the instruction is a Branch instruction, the PC will jump to the appropriate address depending on the ALU flags. Any other instruction, the PC will be incremented by 4. (We still did not implement FENCE, ECALL, EBREAK)
- The Data Memory has as input the instruction's funct3 which decides the size (byte, half-word, word) to load or store. Reading or writing depends on MemRead and MemWrite controls.

- Controls:

- Main Control Unit controls:

Type	**ImmSel**	**MemRead**	**MemWrite**	**ToReg**	**ALUOP**	**ALUSrc1**	**ALUSrc2**	**RegWrite**
Arith-R	0	0	0	01	10	0	0	1
Load	0	1	0	10	00	0	1	1
Store	0	0	1	00	00	0	1	0
Branch	0	0	0	00	01	0	0	0
LUI	0	0	0	01	11	0	1	1
AUIPC	0	0	0	11	00	1	1	1
JAL	0	0	0	00	00	1	1	1
JALR	0	0	0	00	00	0	1	1
Arith-I	0	0	0	01	10	0	1	1

* Please note that ImmSel and ALUSrc1 have no functionality and they were added based on old design. They are left in the controls because if we decide to revert back to old design in later stages.

- ALU Control Unit:

ALUOP	**ALUSelection**	**Operation**
00	0000	Add
01	0001	Subtract
10	Depending on opcode, funct3, funct7	Depending on instruction
11	0011	Pass input to output

- ALU Operations:

ALUSelection	**Operation**
0000	Add
0001	Subtract
0011	Pass (Propagate)

0100	Or
0101	And
0111	Xor
1000	Shift Right Logical (srl)
1001	Shift Left Logical (sll)
1010	Shift Right Arithmetic (sra)
1101	Set less than (slt)
1111	Set less than unsigned (sltu)

- Branch Control Unit:

Instruction	**PCSelection**	**Output of PC MUX**
Jal	10	PC target adder
Jalr	01	ALU Result
Branch	If successful → 10 Else → 00	PC target adder PC+4
Every other instruction	00	PC+4

- Write Data MUX:

ToReg Control	**Output MUX**
00	PC+4
01	ALU Result
10	Data Memory Output
11	PC target adder