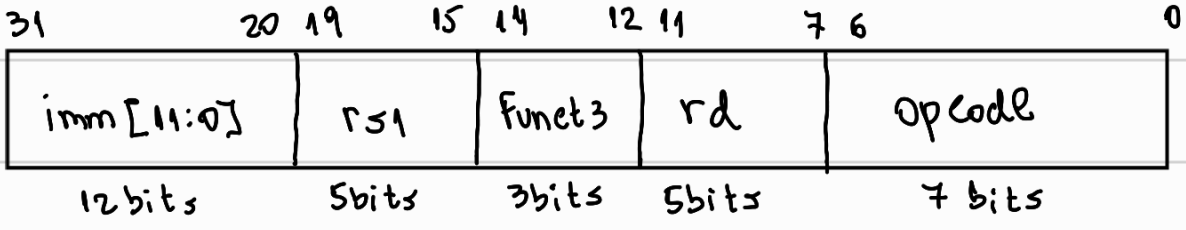
Assignment #2 : Micro Architecture - Single cycle datapath

DANIEL STULBERG HUF

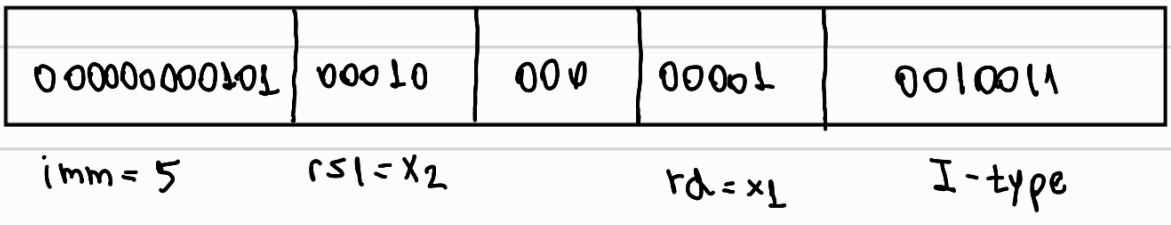
CentraleSupélec

③ ADDI instruction

ADDI: I- type instruction



Example → addi x1, x2, 5



- Steps :

> Read x2 register operands

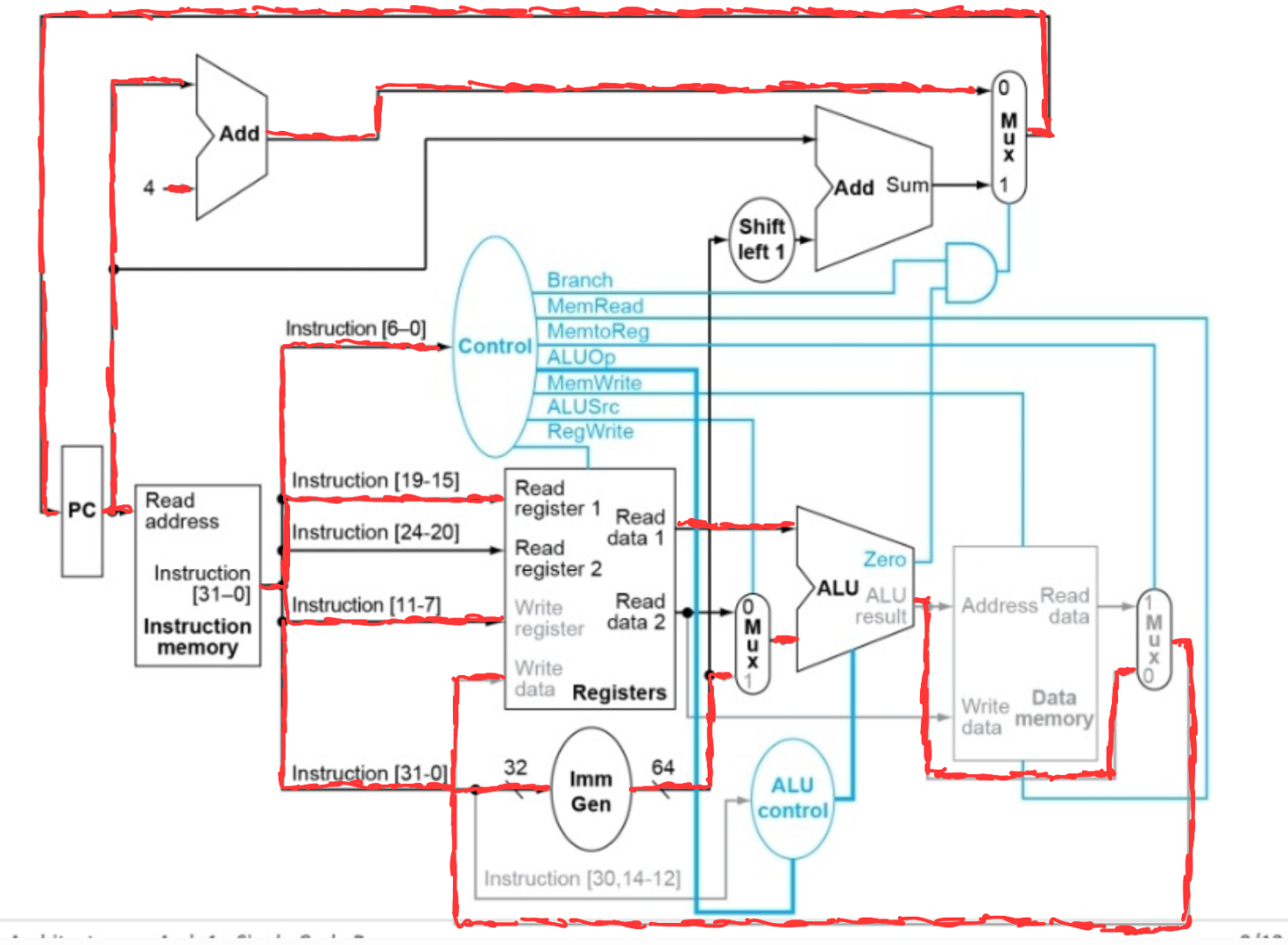
> Highest 12 bits of instruction (inst[31:20]) copied to lowest 12 bits of immediate (imm [11:0])

- Immediate is sign-extended by copying value of inst[31] to fill the upper 20 bits of immediate value (imm [31:12])
- Perform arithmetic /logical operation
- Write register result
- Update PC
  - Operation :

    Reg[rd] <- Reg[rs1] + sign_extend(imm)

    PC = PC+4

  - Datapath:



- Control signals setting

RegWrite : 1

ALUSrc : 1

MemWrite : 0

MemRead : 0

ALUOp1 : 0 } add

ALUOp0 : 0 }

MemtoReg : 0

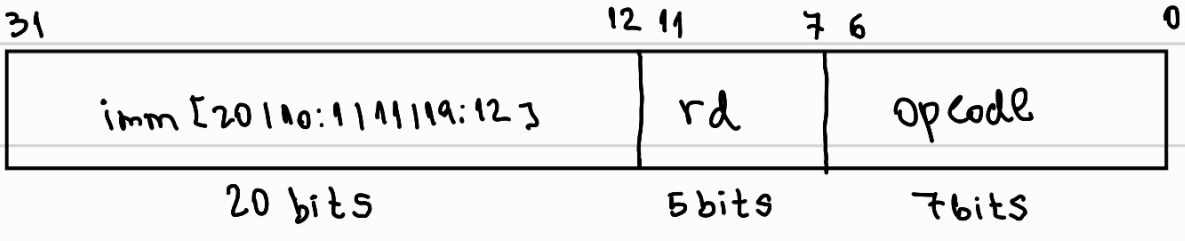
PCSrc : 0

Branch: 0

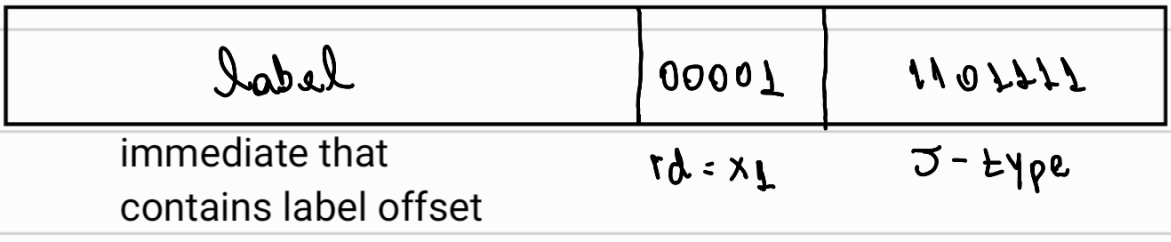
---

① JAL instruction

JAL → J-type instruction

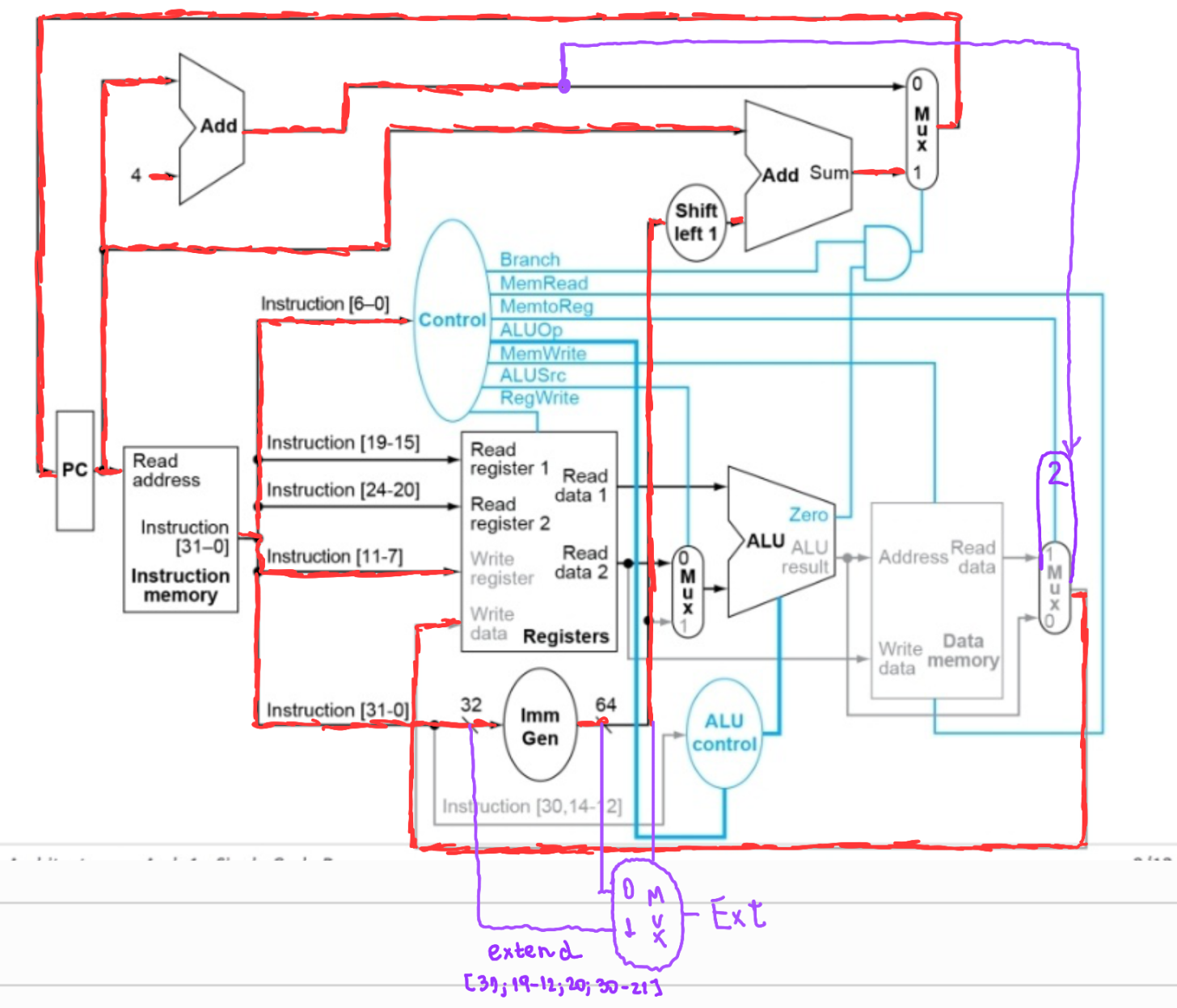


Example ⟶ jal x1, label



- Operation:
  - Reg[rd] <- PC + 4
  - target_address <- PC + 2 x (sign_extend (label))
  - PC = target_ address
    - ↳ 1-bit left shift
- Steps:
  - > place return address in rd
  - > jump to address

- Datapath:



It was added a new wire in order to directly connect PC + 4 to be the input of "write data" by extending the "MemtoReg" mux. I also added another mux that will extend the immediate generated value to 32 bits.

- Control signals setting:

RegWrite : 1

ALUSrc : X

MemWrite : 0

MemRead : 0

ALUOp1 : x

ALUOp0 : x

MemtoReg : 10_2

PCSrc : 1

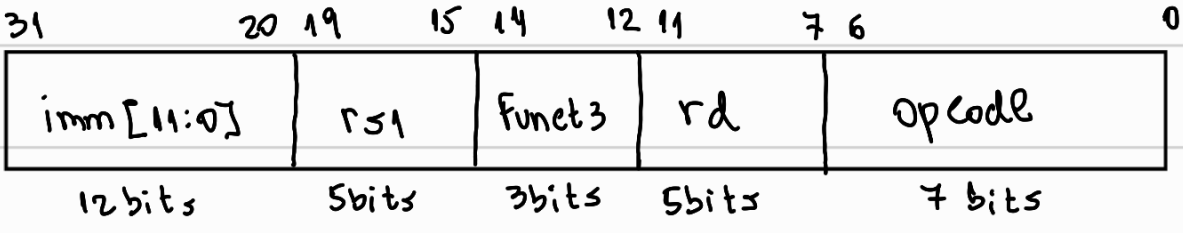
Ext: 1

Branch: x

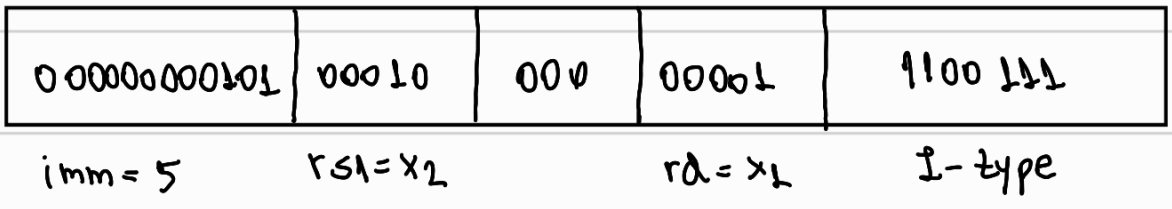
---

(2) JALR instruction

JALR -> I-type instruction



Example → jalr x1, x2, 5



- Steps: > Update return address with PC+4

> Read rs1 register operands

> Calculate address adding rs1 with 12-bit offset and place 0 in the least significant bit

of the ALU result.

› Update PC (jump to address)

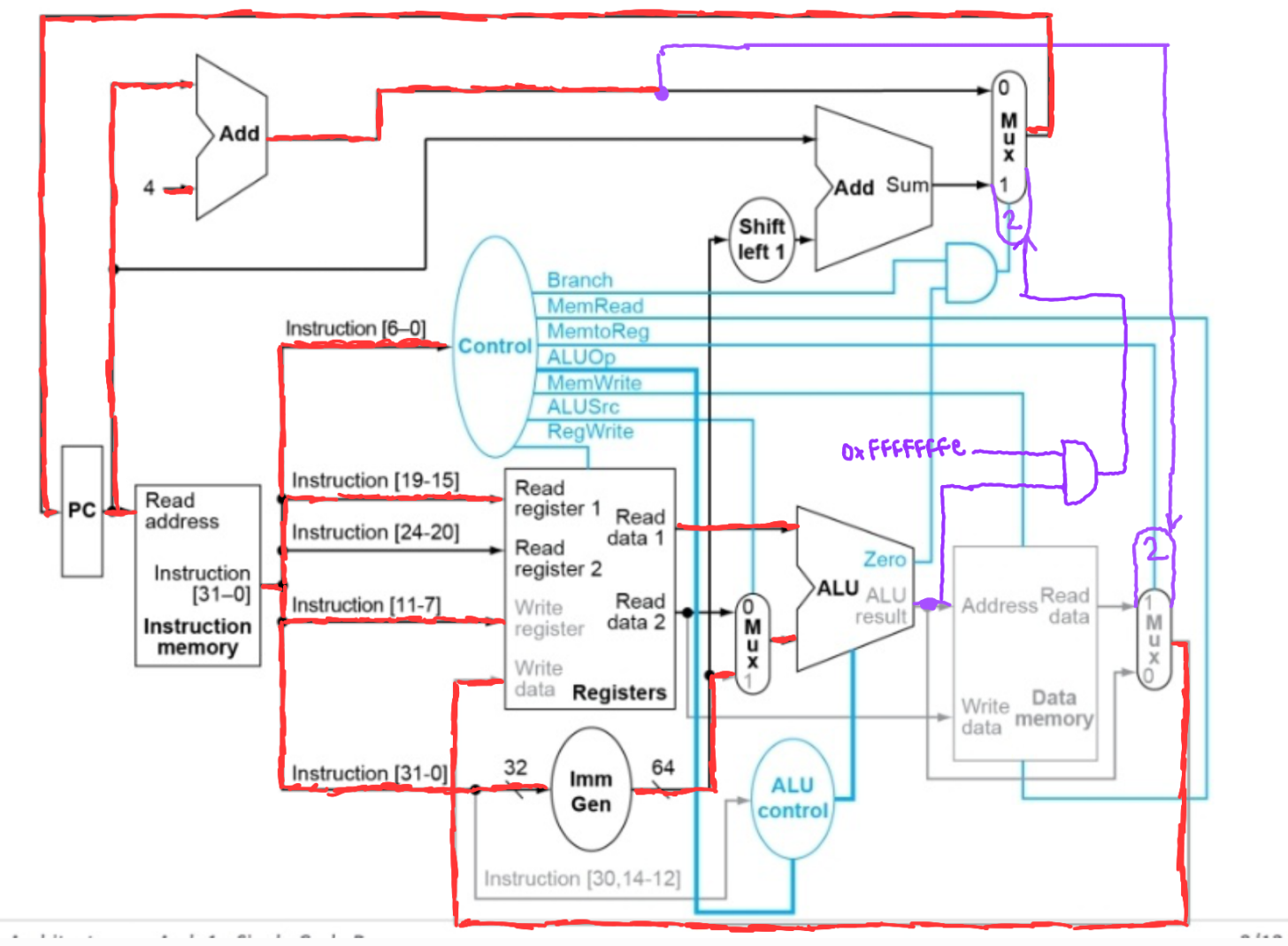
- Operation:

Reg[rd] <- PC+4

target_address <- (Reg[rs1] + sign_extend(imm)) & ~1

PC <- target_address

- Datapath :



An AND port was added in order to turn the least significant bit of the ALU output to 0. In addition to the extension of the "MemtoReg" mux, the "PCSrc" mux was also extended to connect the AND port result directly to the PC input.

- Control signals setting

RegWrite : 1

ALUSrc : 1

MemWrite : 0

MemRead : 0

MemtoReg : 10_2

ALUOp 1 : 0 } add

ALUOp 0 : 0 } add

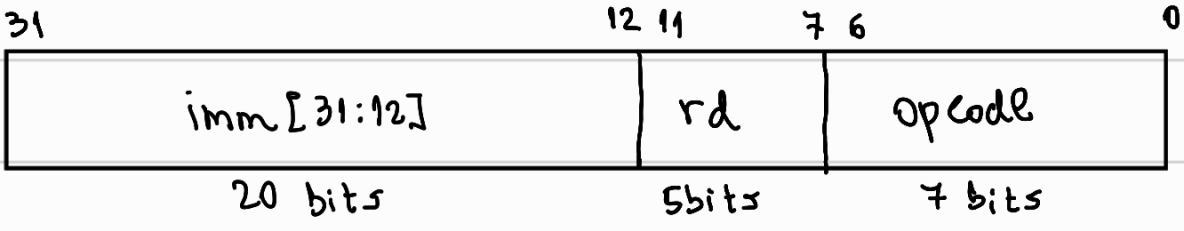
PCSrc : 10_2

Branch : 0

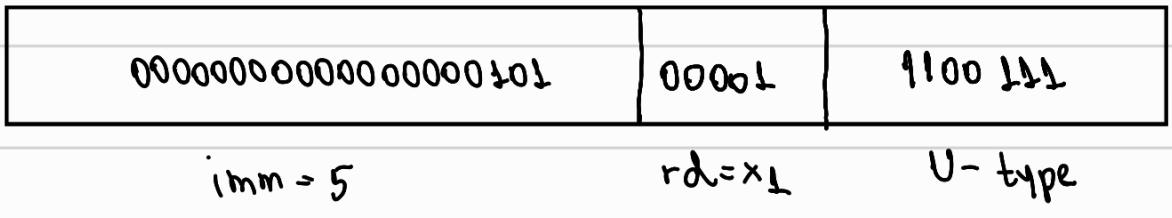
---

4 LUI instruction

LUI → U-type instruction



Example → lui x1,5



- Steps:

> Immediate value is placed in the top 20 bits of register rd, filling in the lowest 12 bits with zeros.

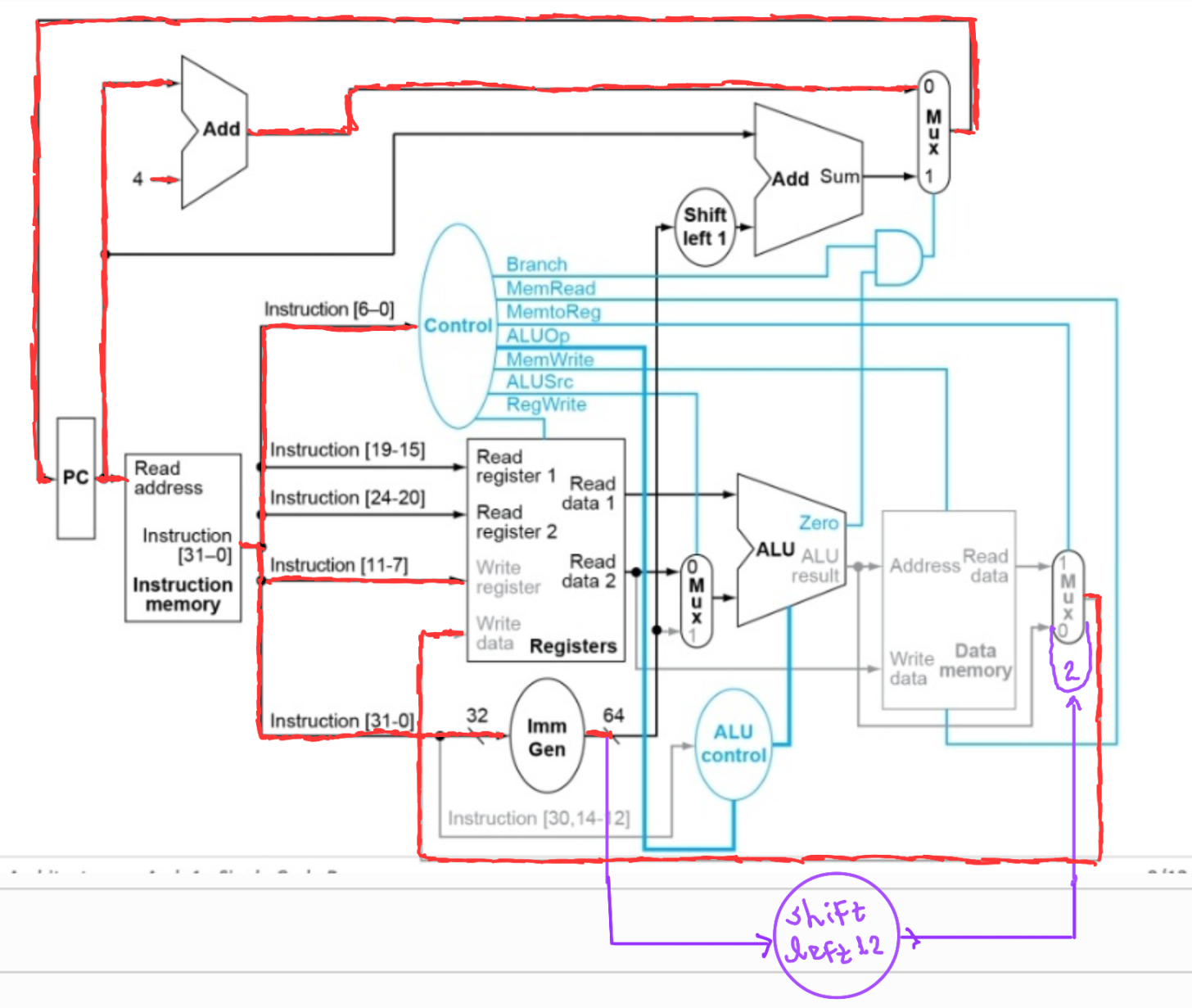
> PC is updated

- Operation:

  Reg[rd] <- sign_extend (imm[31:12] << 12)

  PC <- PC+4

- Datapath:



A block was added after the immediate generator, whose function is shift left the generated value by 12 bits. The "MemtoReg" mux was extended again to receive the output of the sign-extended shifted immediate value, who will be written in the write register.

- Control signals setting

RegWrite : 1

ALUSrc : X

MemWrite : 0

MemRead : 0

ALUOp1 : X

ALUOp0 : X

MemtoReg : 10_2

PCSrc : 10_2

Branch: 0