

PC (PROGRAM COUNTER) - contains the address of the instruction that will be executed. It stores 32 bits / 4 bytes.

CLOCK - This is a single-clock architecture. We execute one instruction in one CK. We use LOAD to start the execution. (1 - ON, 0 - OFF)



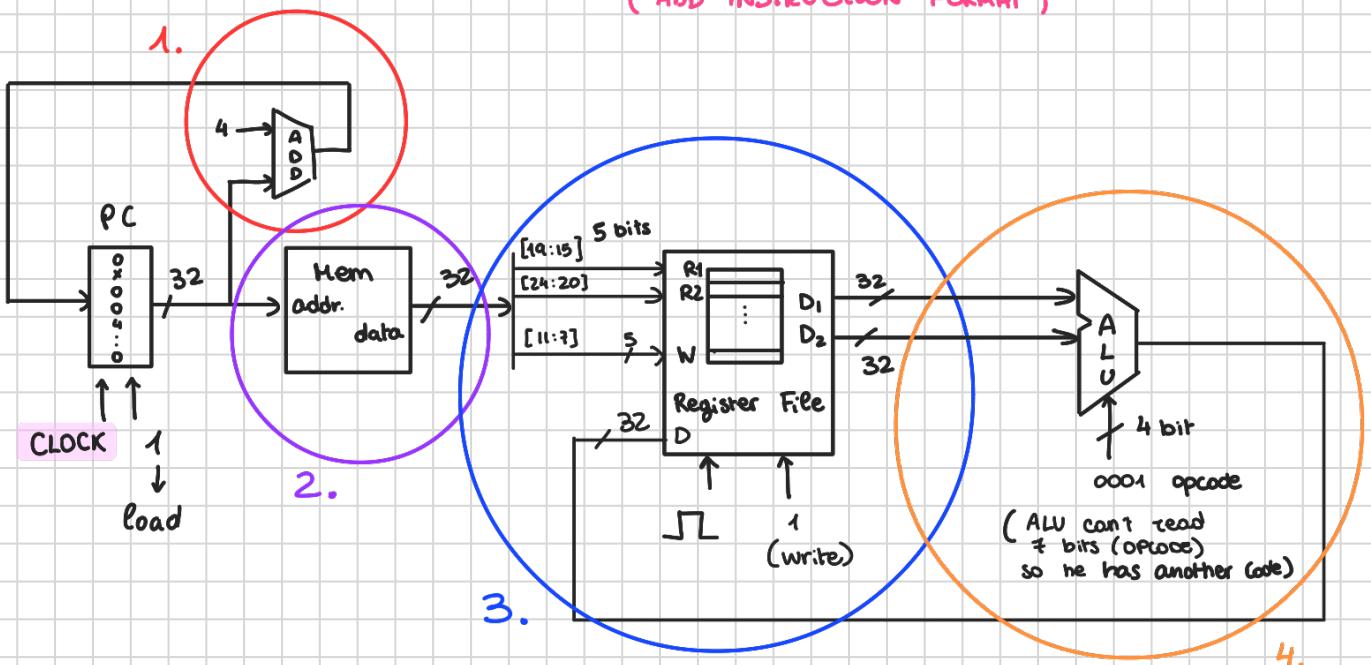
LET'S SEE HOW TO EXECUTE AN ADD INSTRUCTION

add x5, x6, x7
(address 0x00400000)

31	25 24	20 19	15 14	12 11	7 6	0	*
0 ... 0	00111	00110	000	00101	0110011	opcode	

$r_s_2 \quad r_s_1 \quad f_3 \quad r_d$

(ADD INSTRUCTION FORMAT)

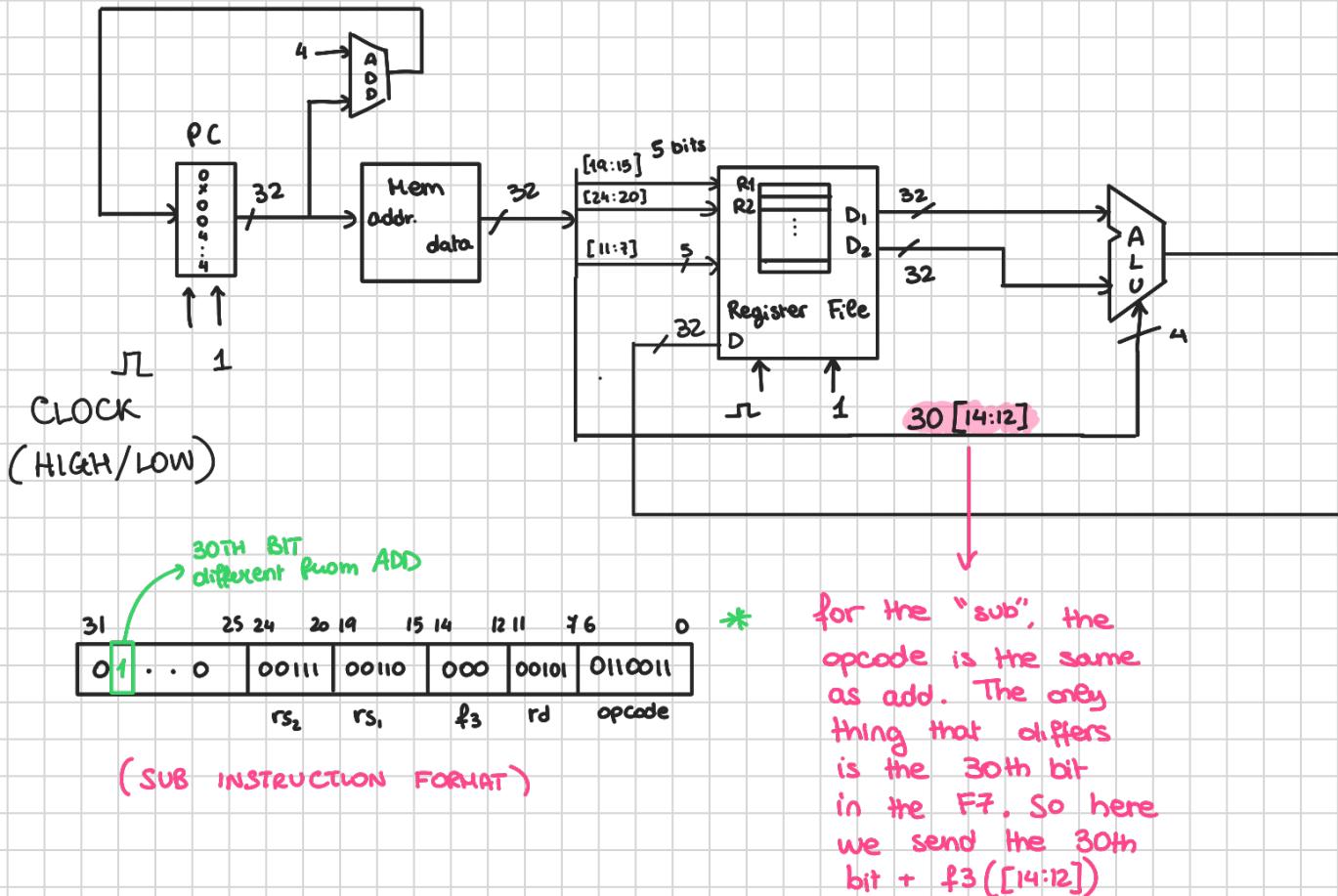


1. So the PC contains the address of the instruction and in order to do the **next instruction**, we will use an adder to increment our address of 4 bytes (+4).
2. Here, we take the address and find the corresponding instruction. We will see the format * of the instruction.
3. Here, we load the addresses of our registers to the CPU. [19:15] is the part of the format where we have r_s_1 . [24:20] // // // // r_s_2 . [11:7] // // // // = r_d . And then, we read their contents using other registers of the CPU and send to the ALU.

4. Now we perform the actual operation (addition).
 The ALU will know what operation to perform thanks to the code sent through the control lines (ALU has 4 control lines, so 4 bits)
 The output will go to the destination register D.

SUB INSTRUCTION

sub x5, x6, x7 (0x00400004)

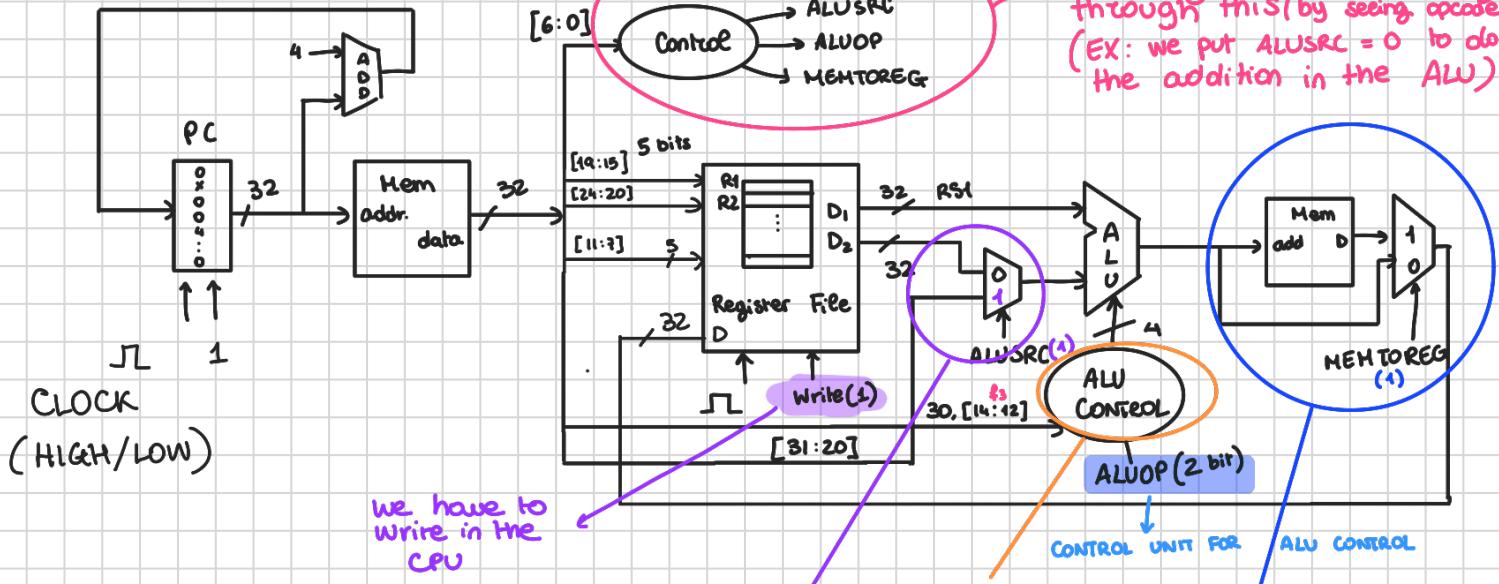


LOAD WORD

lw rd, offset(rs1)

31	20 19	15 14	12 11	7 6	0
IMMEDIATE	00110	000	00101		

rs f₃ rd opcode



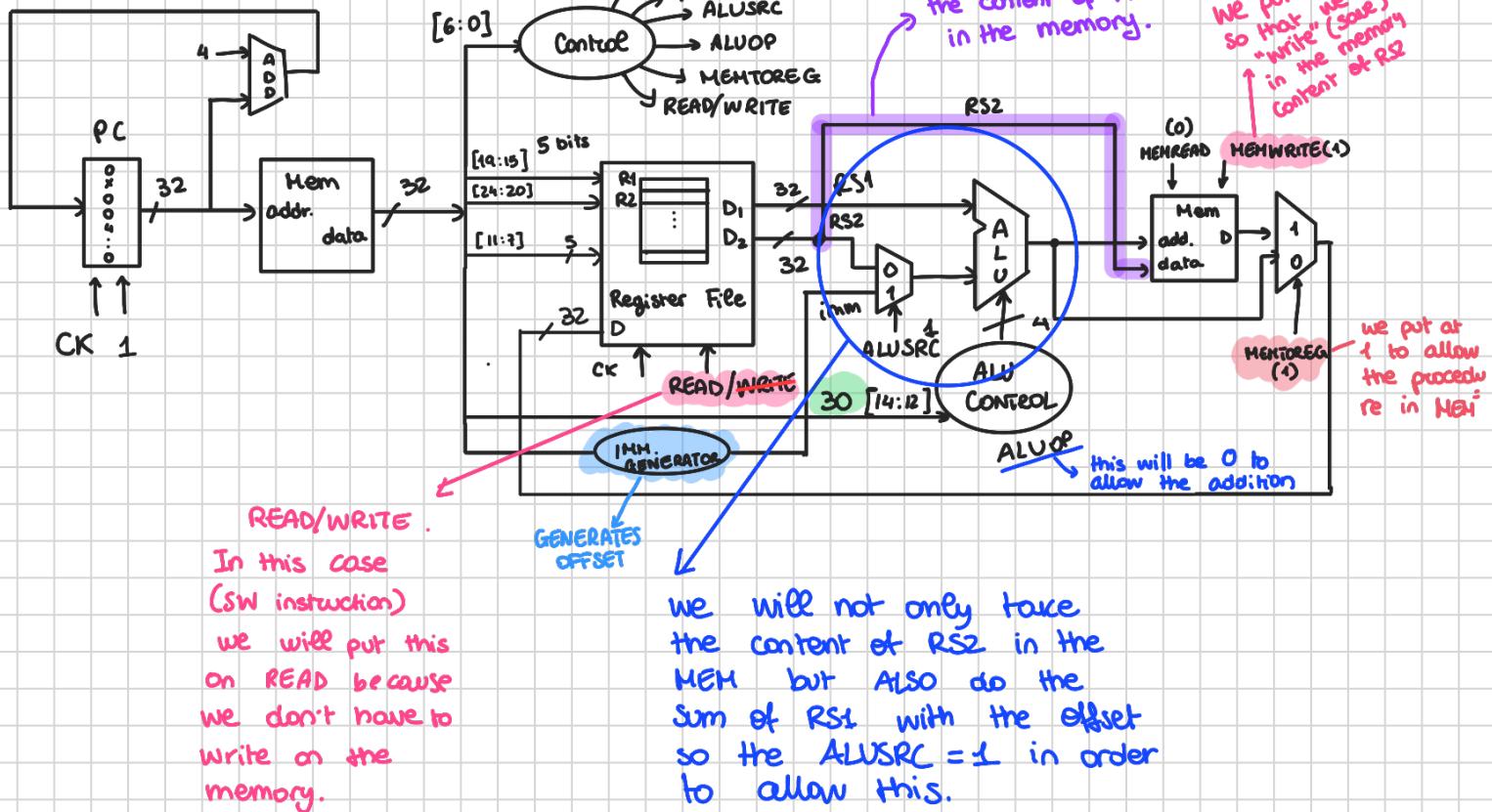
OFFSET. This way we add it to rs1 ([31:20]). We put the ALUSRC to 1 so we can select the second input with the offset.

opcode isn't same so we use a general ALU control

Here, we get the address, take the content and put it into our destination register.

To do so, we have to put MEMTOREG = 1 so we can select with the MUX (multiplexer)

SW RS2, offset(rs1)



VALUES THEY SHOULD BE PUT AT FOR EACH INSTR.

ADD

- ALUSRC = 0
- ALUOP = 0
- MEMTO REG = 0
- READ/WRITE = WRITE
- MEMREAD = 0
- MEMWRITE = 0

SUB

- ALUSRC = 0
- ALUOP = 1
- MEMTO REG = 0
- READ/WRITE = WRITE
- MEMREAD = 0
- MEMWRITE = 0

LW

- ALUSRC = 1
- ALUOP = 0
- MEHTO REG = 1
- READ/WRITE = WRITE
- MEMREAD = 1
- MEMWRITE = 0

SW

- ALUSRC = 1
- ALUOP = 0
- MEHTO REG = 1
- READ/WRITE = READ
- MEMREAD = 0
- MEMWRITE = 1