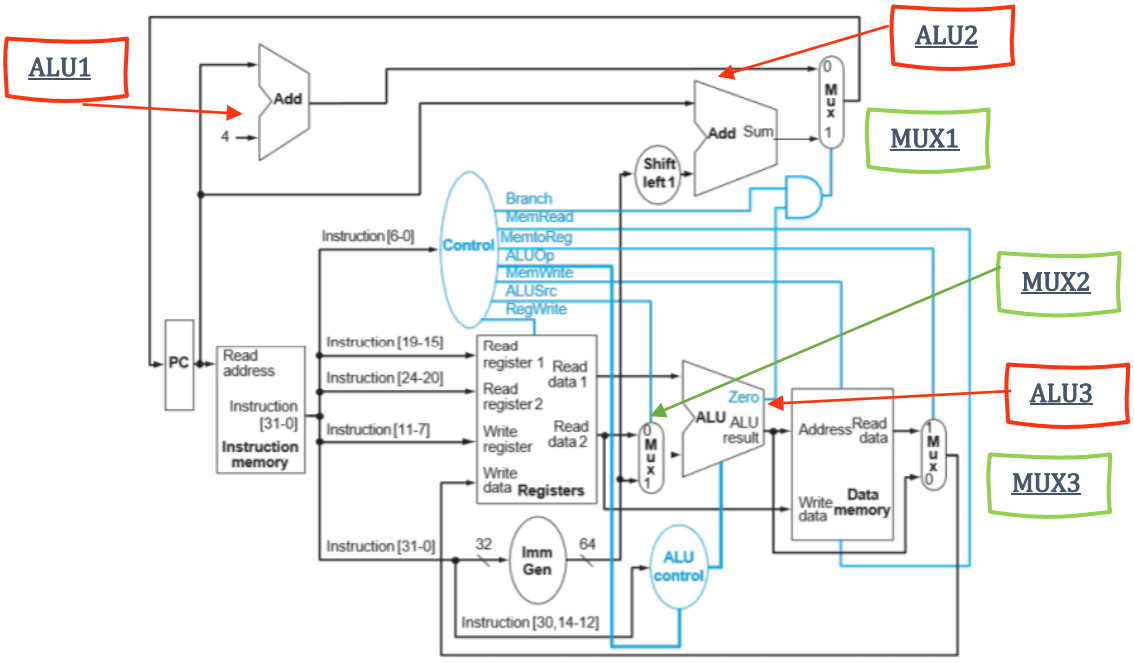
OPTIMITZACIÓ RISC-V

1.5 En aquest exercici, examinarem detalladament com s'executa una instrucció en un "datapath" d'un cicle únic (single cycle). Els problemes d'aquest exercici fan referència al cicle de rellotge en què el processador obté la següent instrucció: 0x00c6ba23 (ISA RV64I).



(a)

Instruction	ALUSrc	Memto-Reg	Reg-Write	Mem-Read	Mem-Write	Branch	ALUOp1	ALUOp0
R-format	0	0	1	0	0	0	1	0
ld	1	1	1	1	0	0	0	0
sd	1	X	0	0	1	0	0	0
beq	0	X	0	0	0	1	0	1

(b)

Figura 2: (a) "Datapath" simplificat amb la unitat de control del RISC-V single cycle.
(b) Taula amb els valors de les línies de control segons tipus d'instrucció.

Primero pasamos la instrucción a binario: **0x00c6ba23** → `1100 0110 1011 1010 0010 0011`

	31 27	26 25	24 20	19 15	14 12	11 7	6 0
R	funct7		rs2	rs1	funct3	rd	Opcode
I	imm[11:0]			rs1	funct3	rd	Opcode
S	imm[11:5]		rs2	rs1	funct3	imm[4:0]	opcode
SB	imm[12|10:5]		rs2	rs1	funct3	imm[4:1|11]	opcode
U	imm[31:12]					rd	opcode
UJ	imm[20|10:1|11|19:12]					rd	opcode

El Opcode serán siempre los 7 bits menos significativos, en nuestro caso: `010 0011`, así que buscamos la instrucción que corresponde a éste y rellenamos las partes según corresponde.

sd rs2, offset(rs1) `M[x[rs1] + sext(offset)] = x[rs2][63:0]`

Store Doubleword. Tipo S, solo RV64I.

Almacena los ocho bytes del registro x[*rs2*] a memoria en la dirección x[*rs1*] + *sign-extend*(*offset*).

Formas comprimidas: **c.sdsp** rs2, offset; **c.sd** rs2, offset(rs1)

31 25	24 20	19 15	14 12	11 7	6 0
offset[11:5]	rs2	rs1	011	offset[4:0]	0100011

La instrucción corresponde a un **sd x12, 20(x13)**

offset[11:5]	rs2	rs1	funct3	offset[4:0]	opcode
0000 000	0 1100	0110 1	011	1010 0	010 0011
	12	13		20	

EC – Qüestionari 3 1.5

1) Quins són els valors de les entrades de la unitat de control ALU per a aquesta instrucció?

* En la tabla rosa de arriba miramos los valores de ALUOp1 y ALUOp0 para nuestra instrucción de tipo sd, que corresponden a **0 0**

ALUOp		Funct7 field							Funct3 field			
ALUOp1	ALUOp0	I[31]	I[30]	I[29]	I[28]	I[27]	I[26]	I[25]	I[14]	I[13]	I[12]	Operation
0	0	X	X	X	X	X	X	X	X	X	X	0010
X	1	X	X	X	X	X	X	X	X	X	X	0110
1	X	0	0	0	0	0	0	0	0	0	0	0010
1	X	0	1	0	0	0	0	0	0	0	0	0110
1	X	0	0	0	0	0	0	0	1	1	1	0000
1	X	0	0	0	0	0	0	0	1	1	0	0001

Las entradas de la unidad de control para nuestros valores son **0010**

2) Quina és la nova adreça del PC després d'executar aquesta instrucció? Ressalteu el camí a través del qual es determina aquest valor (figura 2).

- If we don't take the branch:
  `PC = PC+4 =` next instruction
- If we do take the branch:
  `PC = PC + (immediate*4)`

PC+4, porque no es un branch

3) Per a cada mux, mostreu els valors de les seves entrades i sortides durant l'execució d'aquesta instrucció. Enumereu els valors que són sortides de registre com Reg [xn] (on n és el nombre de reg).

MUX 1
Salida: PC+4
Entradas: PC+4
PC+Shift Left (PC+40)

Shift Left = x2
20x2 = 40

La salida la determina si es una instrucción de Branch o no. Como nuestra instrucción no lo es, la salida es PC+4

MUX 2
Salida: 20 (valor immediato)
Entradas: rx12
20 (valor immediato)

La salida depende del bit **ALUSrc** en la tabla rosa. Para nuestra instrucción corresponde a un 1, por lo que la salida será la entrada 1: 20

MUX 3
Salida: x (no importa)
Entradas: x (read data pero no se lee nada)
rx13+20 (ALU result)

La salida depende del bit **MemtoReg** en la tabla rosa. Para nuestra instrucción corresponde a una x, por lo que no nos importa y la salida será x.

4) Quins són els valors d'entrada per a l'ALU i les dues unitats addicionals?

ALU 1
Salida: PC+4
Entradas: PC
4

ALU 2
Salida: PC+Shift Left (PC+40)
Entradas: PC
Shift Left 20 (20x2=40)

ALU 3
Salida: rs1+20
Entradas: rx13 (rs1)
20 (immediato)