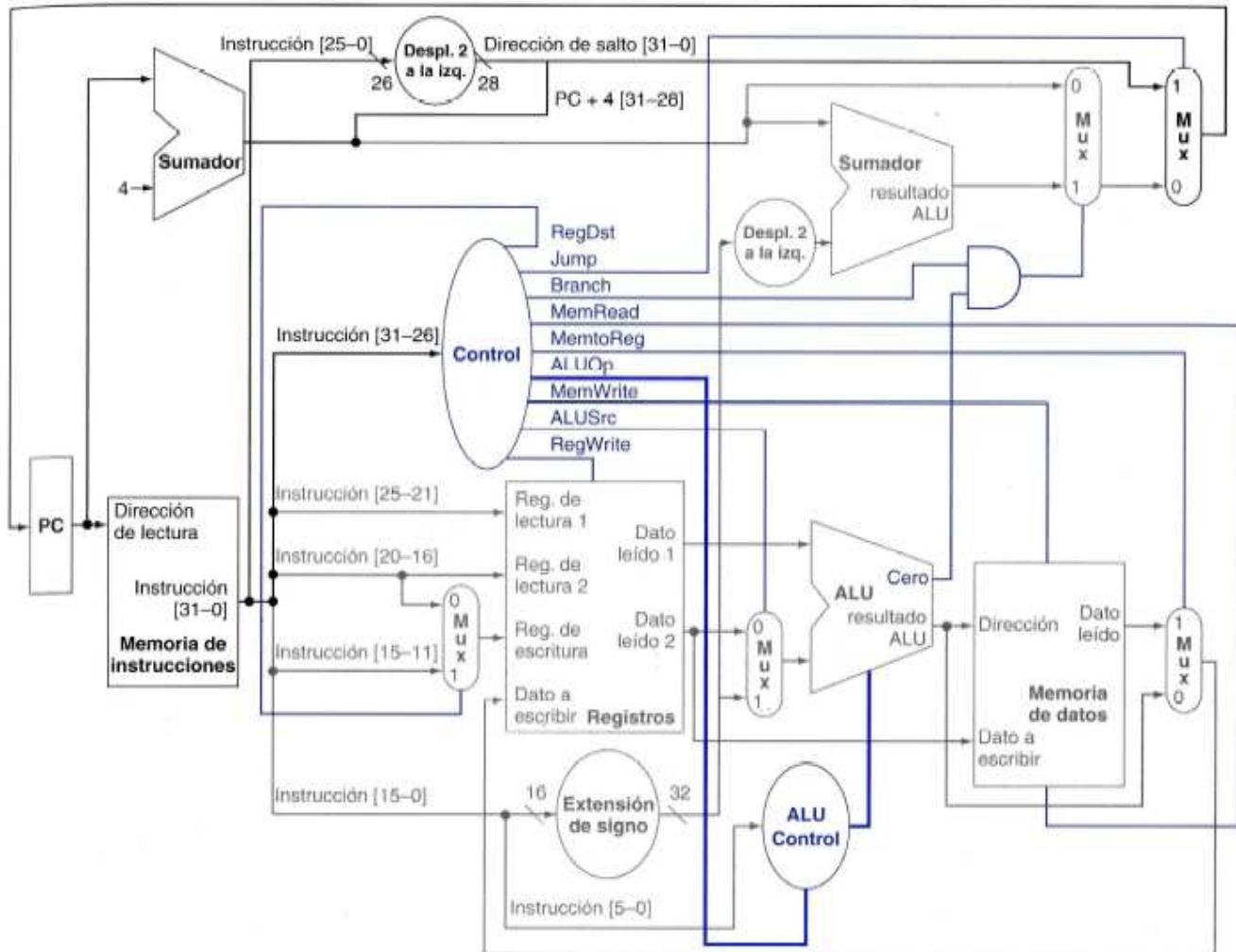


PROBLEMAS TEMA 5. SEGMENTACIÓN Y RIESGOS (RELACIÓN PARA SU EVALUACIÓN)

1)

- a) Camino de datos y señales de control necesarias para incorporar la instrucción *jalr* (jump and link mediante registros, rd = PC y PC = rs, es decir la dirección de retorno PC se guarda en el registro rd, y la dirección contenida en el registro rs se carga en el registro PC) para el camino de datos monociclo realizando las modificaciones en la siguiente figura:

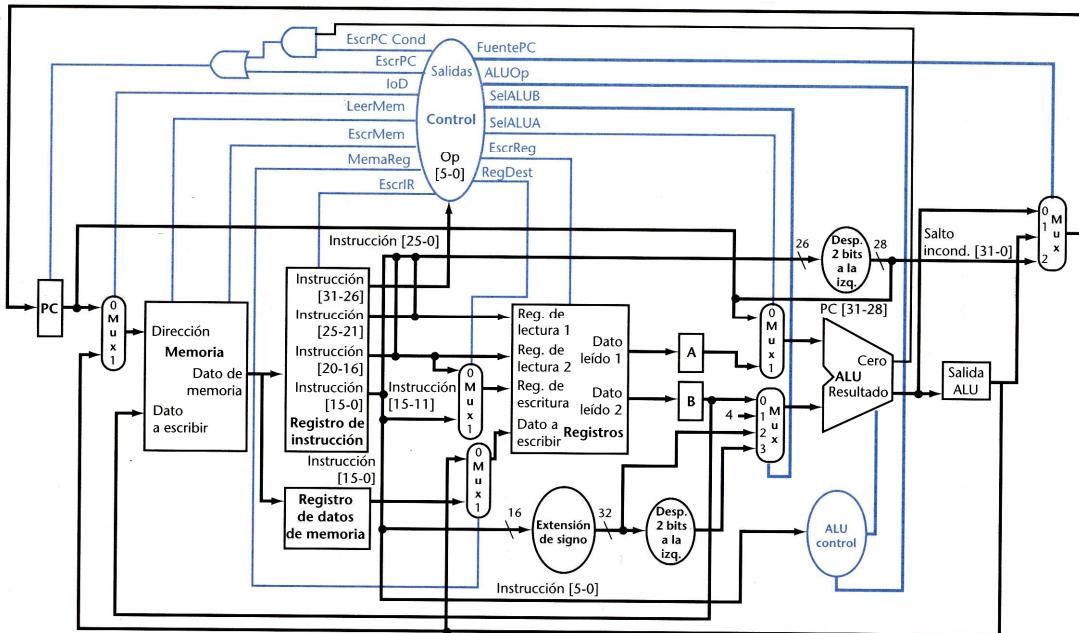


- b) Ampliar la tabla de líneas de control para ver los valores que deben presentar todas las líneas de control que se añadieron en el anterior ejercicio para la instrucción *jalr*.

Instrucción	RegDst	ALUSrc	MemtoReg	RegWrite	MemRead	MemWrite	Branch	ALUOp1	ALUOp0
Formato R	1	0	0	1	0	0	0	1	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	X	0	0	0	1	0	1

2)

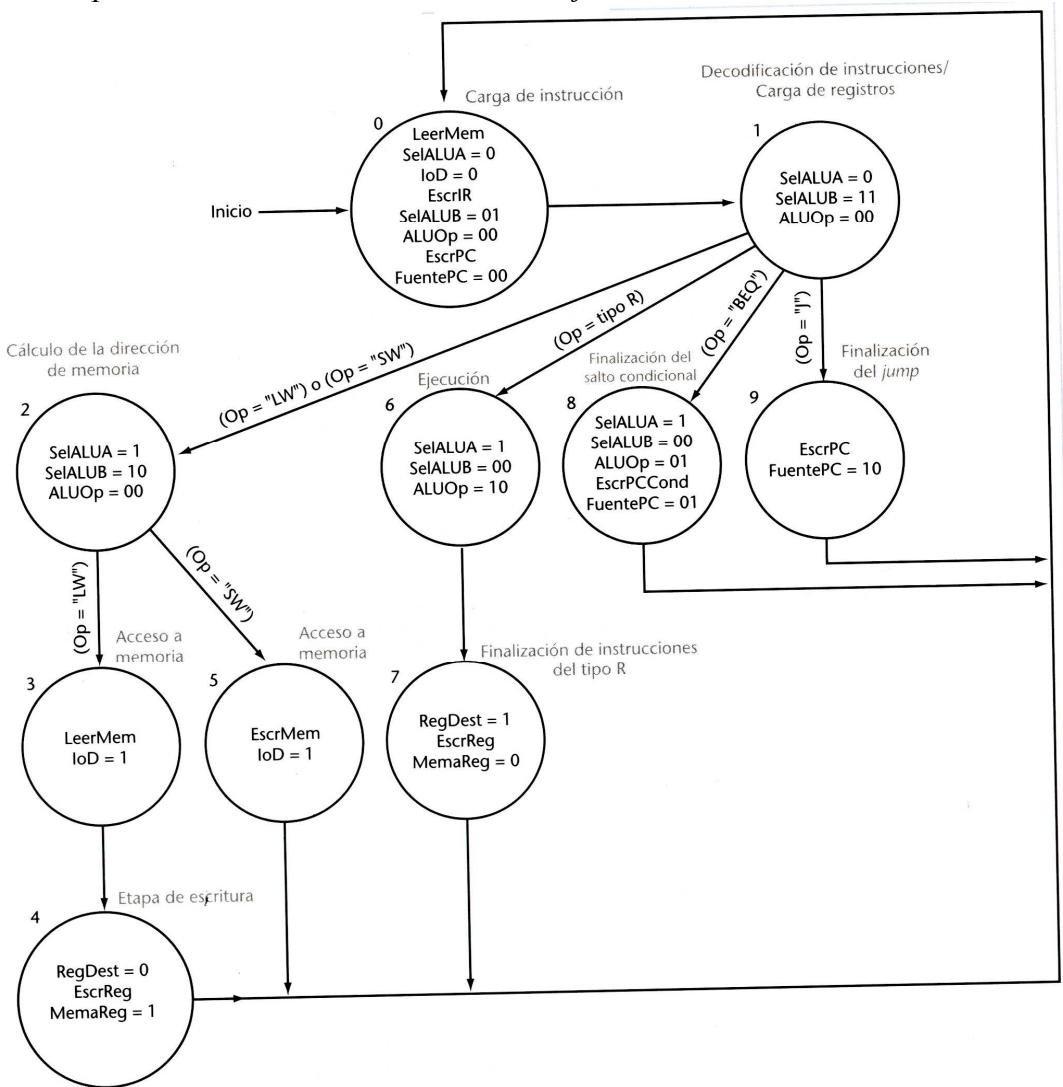
- a) Camino de datos y señales de control necesarias para incorporar la instrucción *jalr* (jump and link mediante registros, rd = PC y PC = rs, es decir la dirección de retorno PC se guarda en el registro rd, y la dirección contenida en el registro rs se carga en el registro PC) para el camino de datos multiciclo realizando las modificaciones en la siguiente figura:



- b) Ampliar la tabla de líneas de control para ver los valores que deben presentar todas las líneas de control que se añadieron en el anterior ejercicio para la instrucción *jalr*.

Nombre de la etapa	Acción para instrucciones del tipo R	Acción para instrucciones de acceso a memoria	Acción para saltos condicionales	Acción para instrucciones Jump
Carga de instrucción		IR = Memoria[PC] PC = PC + 4		
Decodificación de instrucciones/carga de los registros		A = Reg [IIR[25-21]] B = Reg [IIR[20-16]] SalidaALU = PC + (extensión-signo (IR[15-0]) << 2)		
Ejecución, cálculo de direcciones y finalización de saltos condicionales/jump	SalidaALU = A op B	SalidaALU = A + extensión-signo (IR[15-0])	si (A == B) entonces PC = SalidaALU	PC = PC [31-28] (IR[25-0]<<2)
Acceso a memoria y finalización de instrucciones de tipo R	Reg [IIR[15-11]] = SalidaALU	Load: MDR = Memoria[SalidaALU] o Store: Memoria[SalidaALU] = B		
Finalización de la lectura de memoria		Load: Reg[IR[20-16]] = MDR		

- c) Mostrar lo que se debe añadir a la máquina de estados finitos de la figura para implementar el control de la instrucción *jalr*



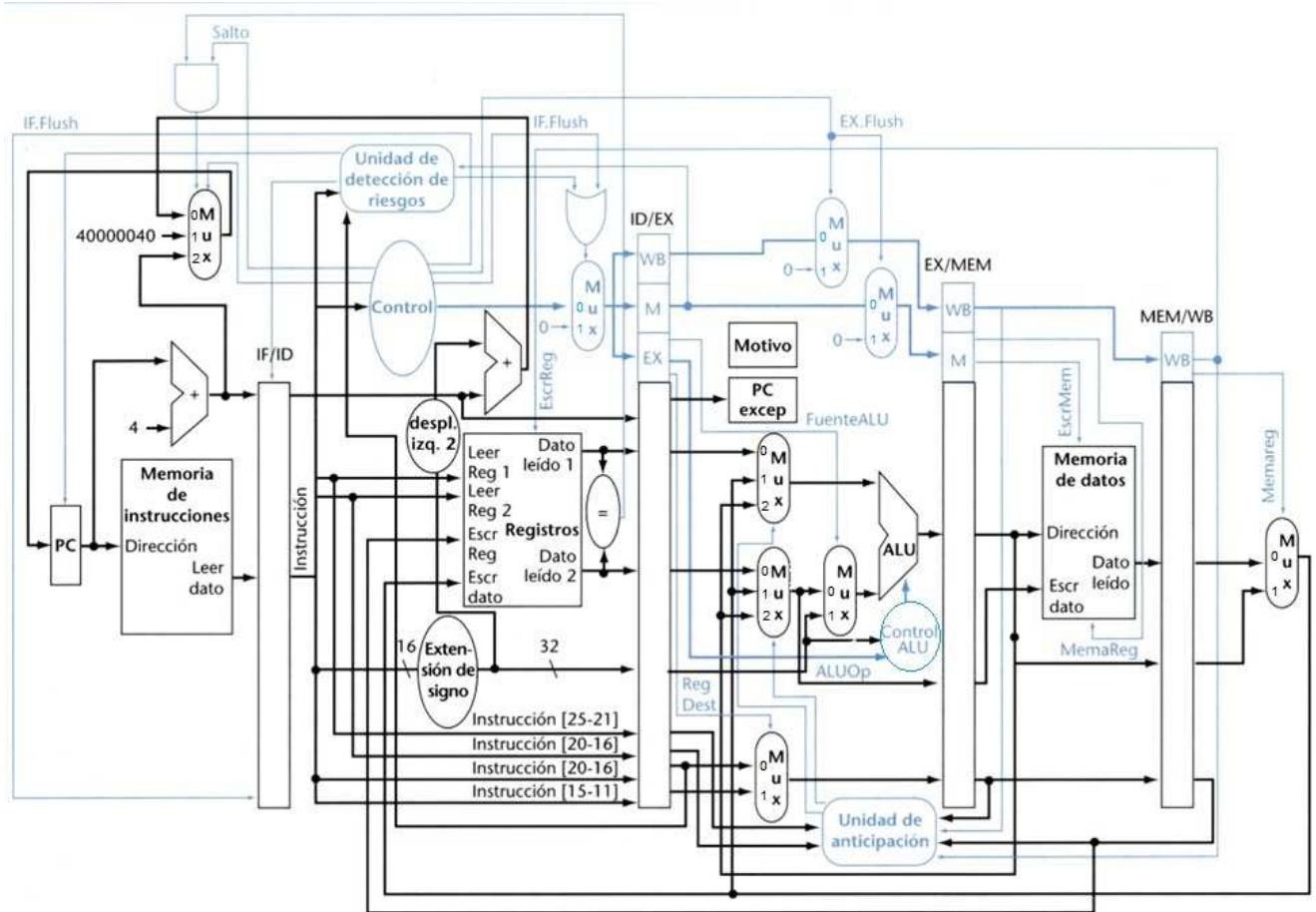
- 3) Las siguientes secuencias de instrucciones hacen uso del camino de datos segmentado del procesador MIPS:

lw \$1,40(\$2) add \$2,\$3,\$3 add \$1,\$1,\$2 sw \$1,20(\$2)	add \$1,\$2,\$3 sw \$2,0(\$1) lw \$1,4(\$2) add \$2,\$2,\$1
Secuencia 1	Secuencia 2

- a) Detectar los posibles riesgos que se producen en las dos secuencias.
- b) Considerando que no hay anticipación ni detección de riesgos, insertar el menor número de *nop* para asegurar una ejecución correcta para cada una de las secuencias.

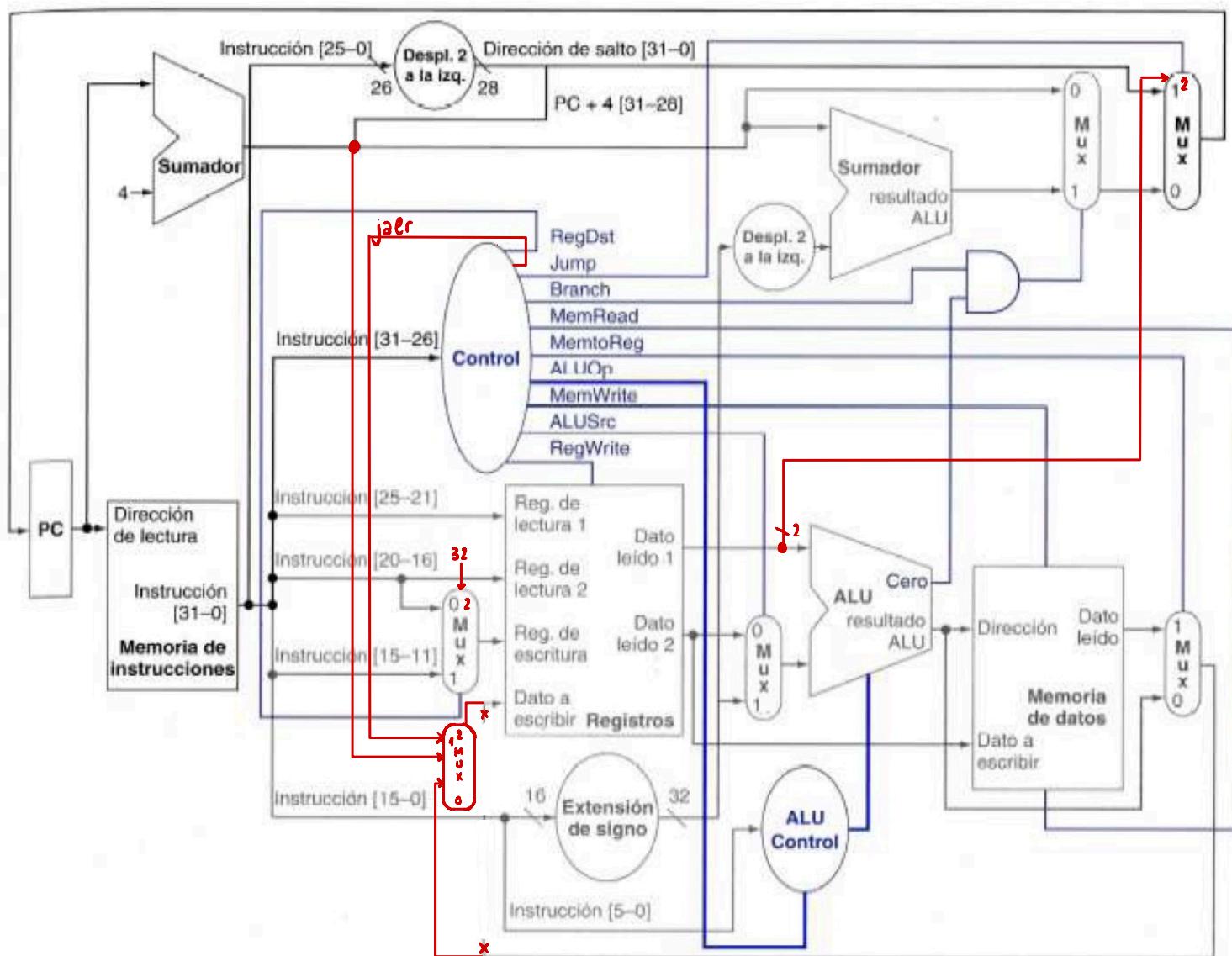
- 4) Determinar en que situación quedan las etapas de segmentación en el ciclo 5 para las tres secuencias de código 1 del ejercicio anterior, y según los siguientes casos que se proponen:
- Con unidad de anticipación y detección, con postescritura y lectura simultánea.

CICLO	ETAPA IF	ETAPA ID	ETAPA EX	ETAPA MEM	ETAPA WB



1)

- a) Camino de datos y señales de control necesarias para incorporar la instrucción *jalr* (jump and link mediante registros, rd = PC y PC = rs, es decir la dirección de retorno PC se guarda en el registro rd, y la dirección contenida en el registro rs se carga en el registro PC) para el camino de datos monociclo realizando las modificaciones en la siguiente figura:

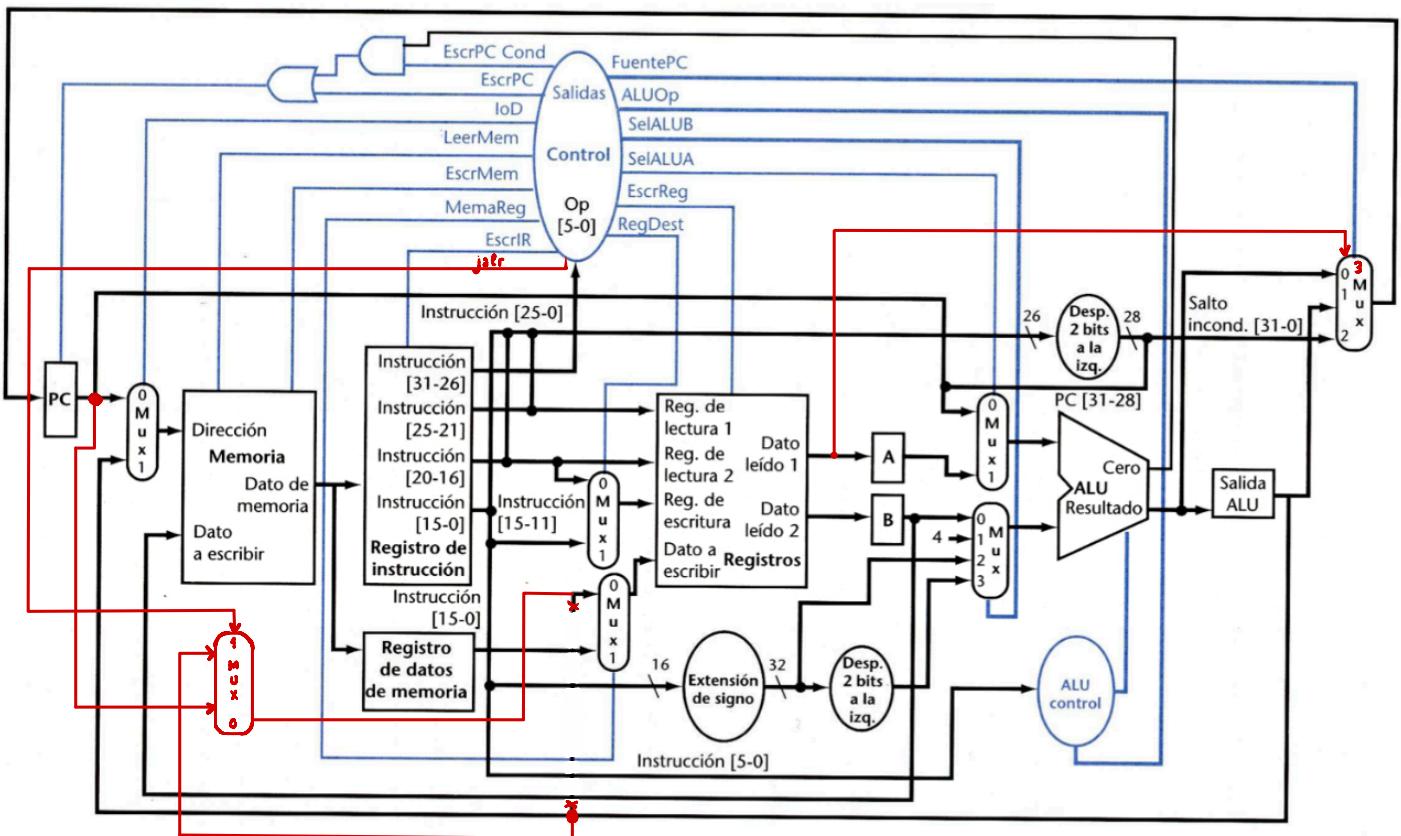


- b) Ampliar la tabla de líneas de control para ver los valores que deben presentar todas las líneas de control que se añadieron en el anterior ejercicio para la instrucción *jalr*.

Instrucción	RegDst	ALUSrc	Memto-Reg	Reg Write	Mem Read	Mem Write	Branch	ALUOp1	ALUOp0	jump	jalr
Formato R	1	0	0	1	0	0	0	1	0	0	0
lw	0	1	1	1	1	0	0	0	0	0	0
sw	X	1	X	0	0	1	0	0	0	0	0
beq	X	0	X	0	0	0	1	0	1	0	0
jalr	1	X	X	1	0	0	0	X	X	10	1

2)

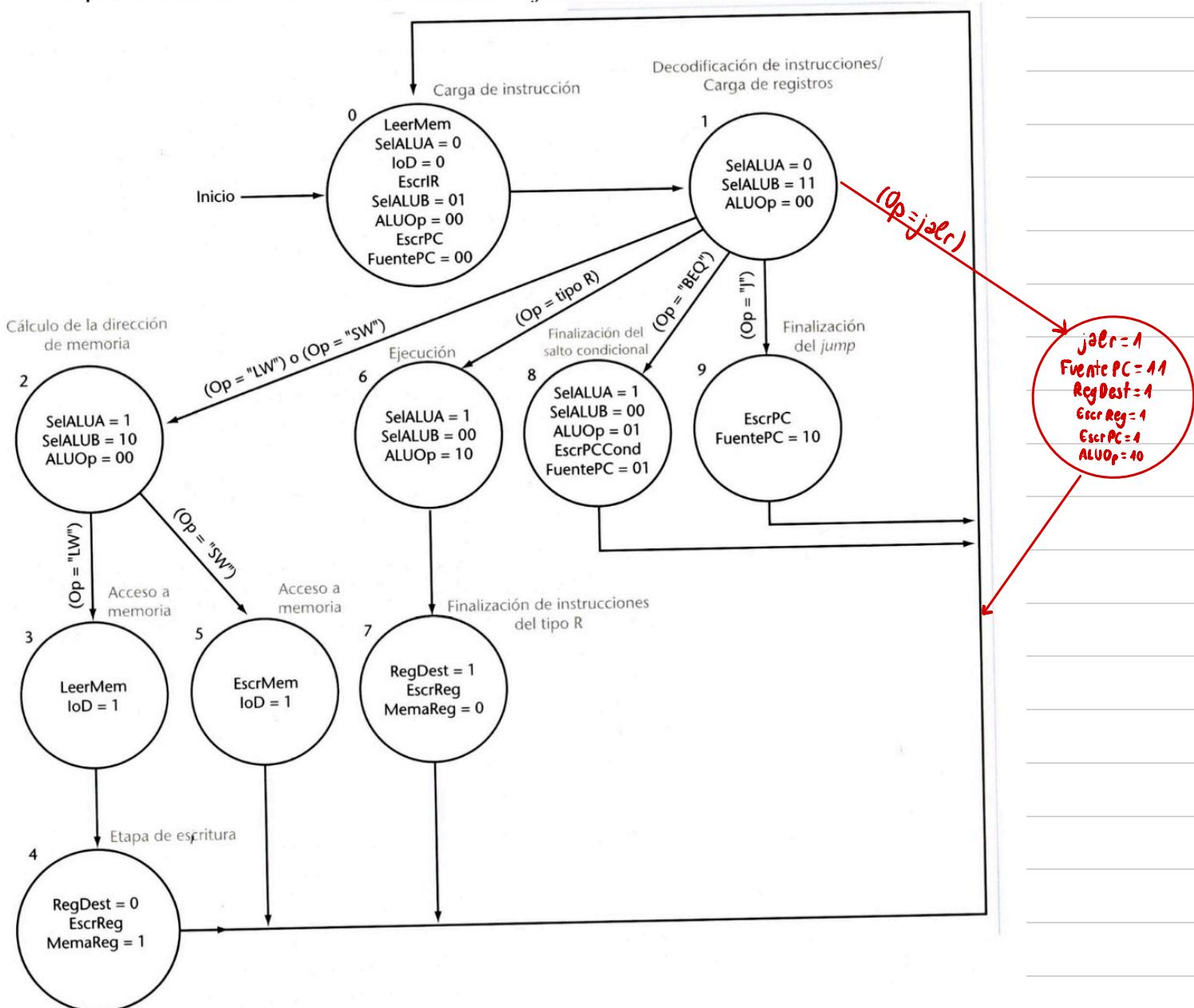
- a) Camino de datos y señales de control necesarias para incorporar la instrucción *jalr* (jump and link mediante registros, rd = PC y PC = rs, es decir la dirección de retorno PC se guarda en el registro rd, y la dirección contenida en el registro rs se carga en el registro PC) para el camino de datos multiciclo realizando las modificaciones en la siguiente figura:



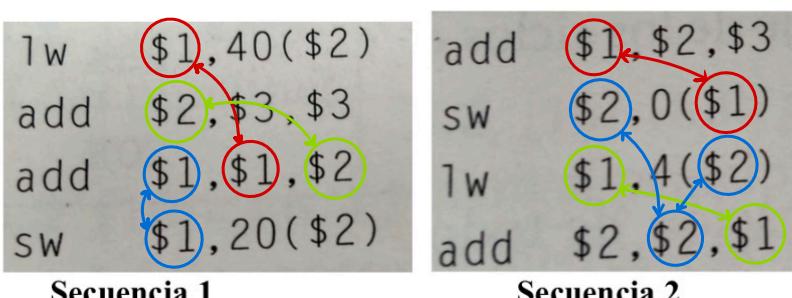
- b) Ampliar la tabla de líneas de control para ver los valores que deben presentar todas las líneas de control que se añadieron en el anterior ejercicio para la instrucción *jalr*.

Nombre de la etapa	Acción para instrucciones del tipo R	Acción para instrucciones de acceso a memoria	Acción para saltos condicionales	Acción para instrucciones Jump	
Carga de instrucción		IR = Memoria[PC] PC = PC + 4			<i>jalr</i>
Decodificación de instrucciones/carga de los registros		A = Reg [IR[25-21]] B = Reg [IR[20-16]] SalidaALU = PC + (extensión-signo (IR[15-0]) << 2)			
Ejecución, cálculo de direcciones y finalización de saltos condicionales/jump	SalidaALU = A op B	SalidaALU = A + extensión-signo (IR[15-0])	si (A == B) entonces PC = SalidaALU	PC = PC [31-28] (IR[25-0]<<2)	<i>Registro [IR[45-14]] = PC+4</i> <i>PC = Registro [IR[25-24]]</i>
Acceso a memoria y finalización de instrucciones de tipo R	Reg [IR[15-11]] = SalidaALU	Load: MDR = Memoria[SalidaALU] Store: Memoria[SalidaALU] = B			
Finalización de la lectura de memoria		Load: Reg[IR[20-16]] = MDR			

- c) Mostrar lo que se debe añadir a la máquina de estados finitos de la figura para implementar el control de la instrucción *jalr*



- 3) Las siguientes secuencias de instrucciones hacen uso del camino de datos segmentado del procesador MIPS:



- a) Detectar los posibles riesgos que se producen en las dos secuencias.
 b) Considerando que no hay anticipación ni detección de riesgos, insertar el menor número de *nop* para asegurar una ejecución correcta para cada una de las secuencias.

SEC 1	Cd/o	If	Id	Ex	Mem	Wb
1	lw					
2	add \$2	lw				
3	add \$1	add \$2	lw			
4	sw	add \$1	add \$2	lw		
5	sw	add \$1	nop	add \$2	lw	
6	sw	add \$1	nop	nop	add \$2	
7	sw	add \$1	nop	nop	nop	
8		sw	add \$1	nop	nop	
9		sw	nop	add \$1	nop	
10		sw	nop	nop	add \$1	
11		sw	nop	nop	nop	
12			sw	nop	nop	
13				sw	nop	
14					sw	

SEC 2	Cd/o	If	Id	Ex	Mem	Wb
1	add \$1					
2	sw	add \$1				
3	lw	sw	add \$1			
4	add \$2	sw	nop	add \$1		
5	add \$2	sw	nop	nop	add \$1	
6	add \$2	sw	nop	nop	nop	
7	add \$2	lw	sw	nop	nop	
8		add \$2	lw	sw	nop	
9		add \$2	nop	lw	sw	
10		add \$2	nop	nop	lw	
11		add \$2	nop	nop	nop	
12			add \$2	nop	nop	
13				add \$2	nop	
14					add \$2	

- 4) Determinar en qué situación quedan las etapas de segmentación en el ciclo 5 para las tres secuencias de código 1 del ejercicio anterior, y según los siguientes casos que se proponen:

a) Con unidad de anticipación y detección, con postescritura y lectura simultánea.

SEC1

CICLO	ETAPA IF	ETAPA ID	ETAPA EX	ETAPA MEM	ETAPA WB
1	lw				
2	add \$2	lw			
3	add \$1	add \$2		lw	
4	sw	add \$1		add \$2	lw
5		sw		add \$1	add \$2
					lw

SEC2 → En el ciclo 6 se usará un nop por \$1 con lw y add \$2

CICLO	ETAPA IF	ETAPA ID	ETAPA EX	ETAPA MEM	ETAPA WB
1	add \$1				
2	sw	add \$1			
3	lw	sw		add \$1	
4	add \$2	lw		sw	add \$1
5		add \$2		lw	sw
					add \$1

