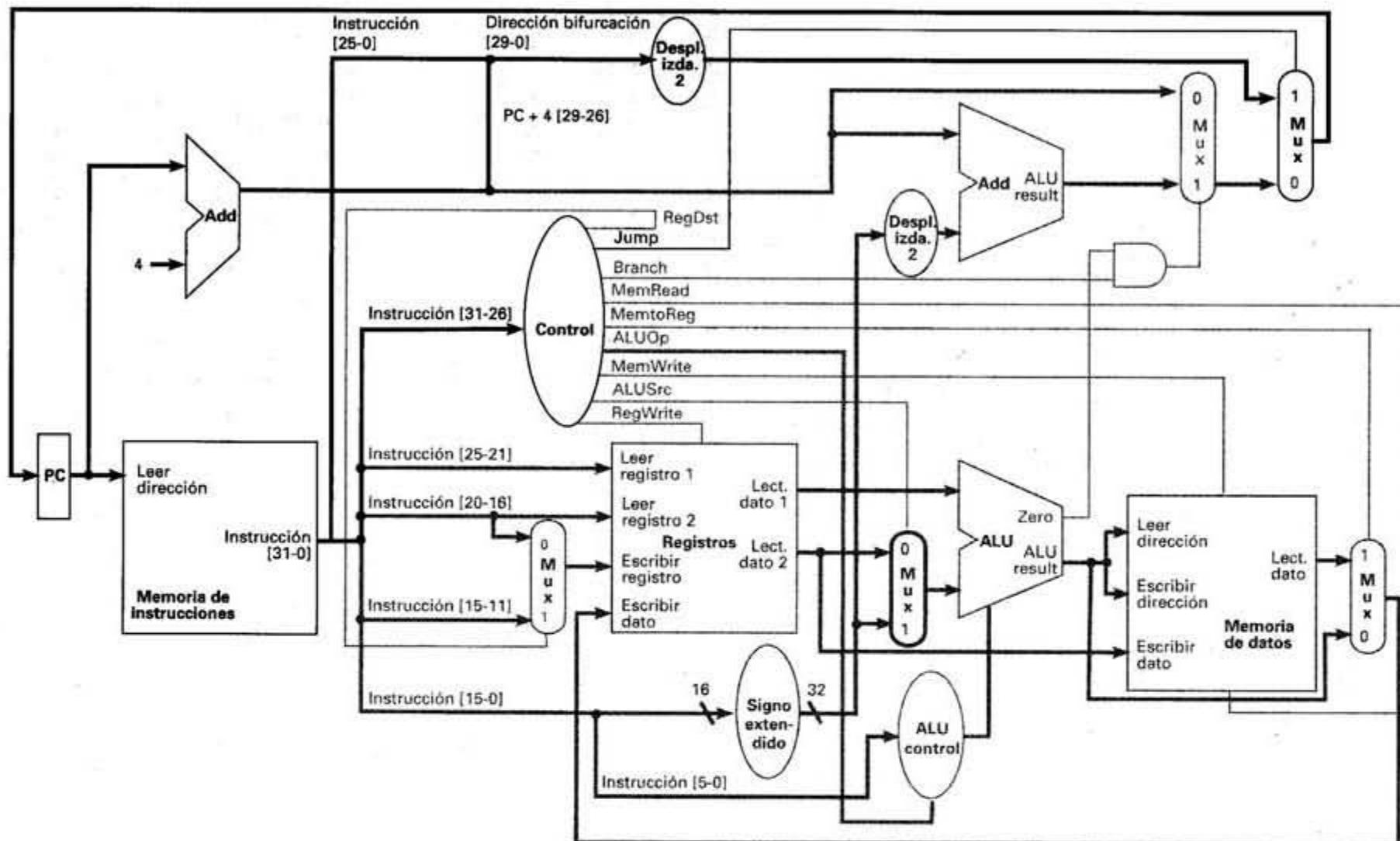


ARQUITECTURAS AVANZADAS DE PROCESADORES

El procesador: Camino de datos y control

HOJA DE PROBLEMAS N° 1

1.- Camino de Datos y señales de Control necesarias para incorporar la Instrucción **jal** (jump and link) para el Camino de Datos Monociclo (las modificaciones se pueden realizar en la figura siguiente):



2.- Ampliar la tabla de inicialización de las líneas de control para ver los valores que deben de presentar todas las líneas de Control que se añadieron en el ejercicio anterior para la Instrucción **jal**:

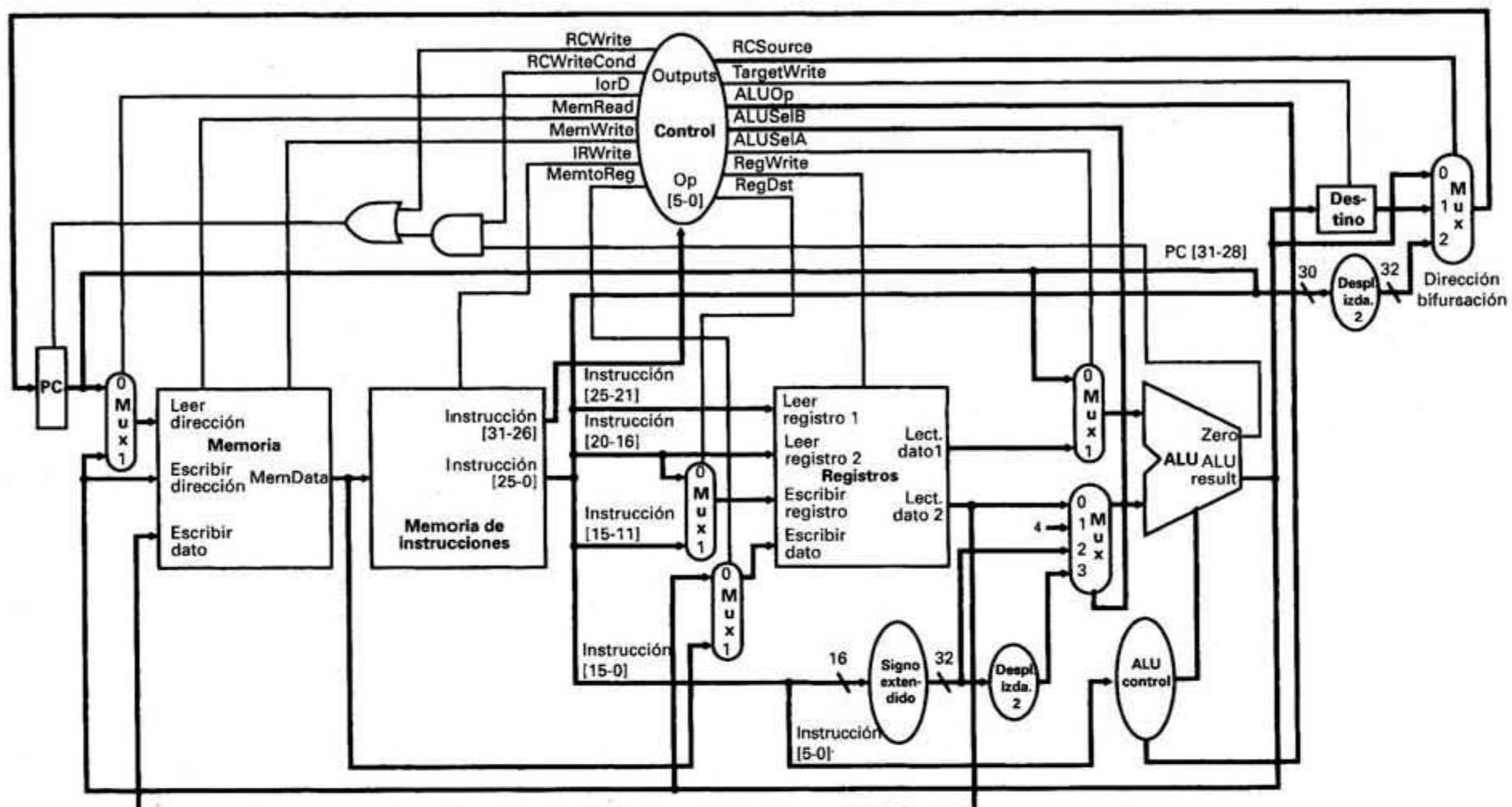
Instrucción	RegDst	ALUSrc	MemtoReg	Reg Write	Mem Read	Mem Write	Branch	ALUOp1	ALUOp0
Formato R	1	0	0	1	0	0	0	1	0
lw	0	1	1	1	1	0	0	0	0
sw	X	1	X	0	0	1	0	0	0
beq	X	0	X	0	0	0	1	0	1

ARQUITECTURAS AVANZADAS DE PROCESADORES

El procesador: Camino de datos y control

HOJA DE PROBLEMAS N° 2

3.- ¿Qué se debe añadir al Camino de Datos y a las Líneas de Control, para que se pueda implementar la Instrucción **jal** en el Camino de Datos Multiciclo, de forma que minimice el número de ciclos de reloj para su ejecución?



4.- Mostrar los pasos en la ejecución de la Instrucción **jal**, en el Camino de Datos Multiciclo, utilizando la misma descomposición de pasos que se muestra en la figura:

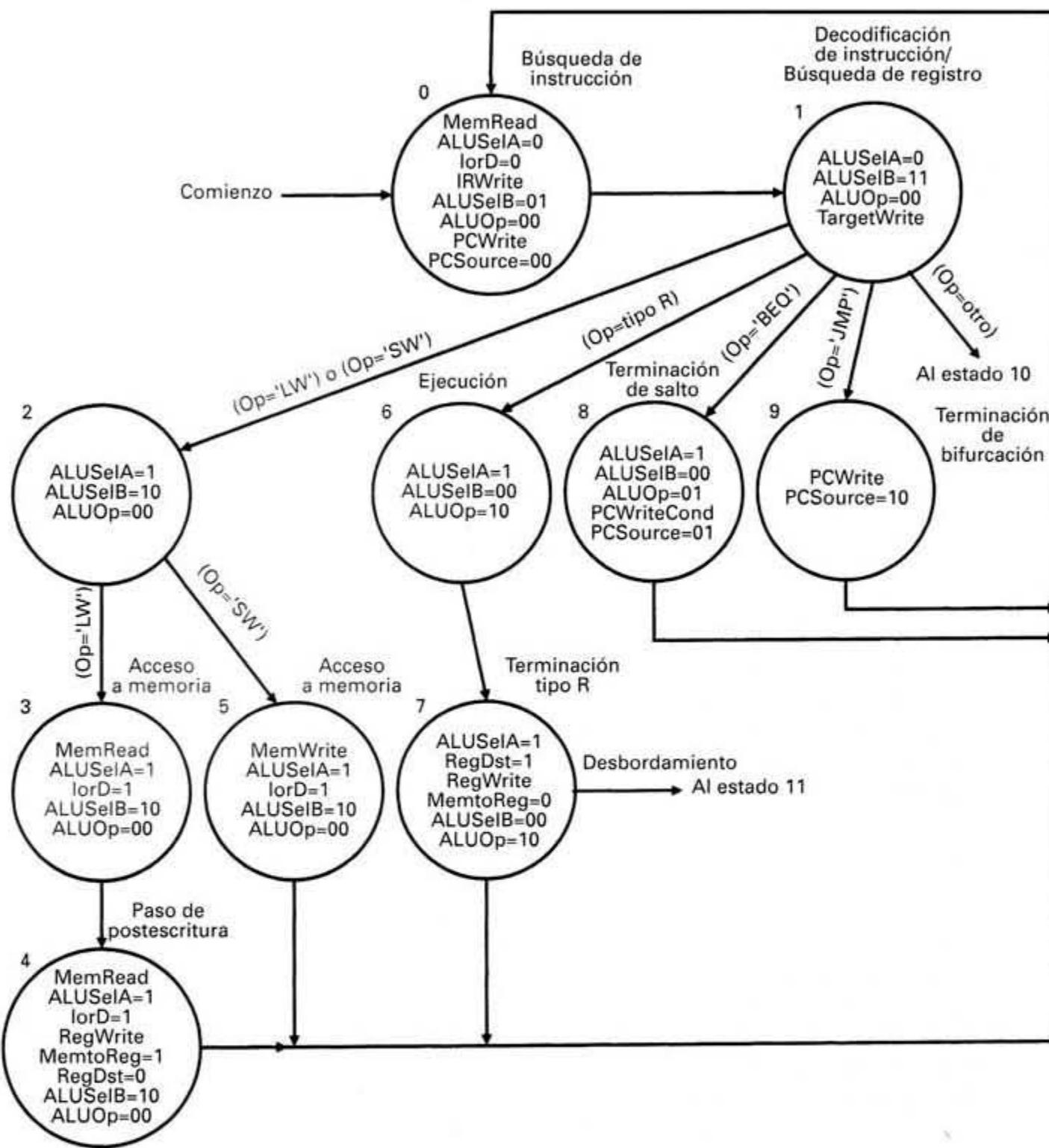
Nombre de paso	Acción para instrucciones tipo R	Acción para instrucciones de referencia a memoria	Acción para saltar
Búsqueda de instrucción		IR=Memoria(PC) PC=PC+4;	
Decodificación de instrucción/Búsqueda de registros		A=Registro[IR[25-21]] A=Registro[IR[20-16]] Target=PC+signo extendido (IR[15-0])<<2)	
Ejecución, cálculo de dirección o terminación de salto	ALUoutput=A op B	ALUoutput=A+signo extendido (IR[15-0])	si (A==B) entonces PC=Target
Acceso a memoria o terminación tipo R	Reg[IR[15-11]]=ALUoutput	dato-memoria=Memoria[ALUoutput] o Memoria [ALUoutput]=B	
Postescritura		Reg[IR[20-16]]=dato-memoria	

ARQUITECTURAS AVANZADAS DE PROCESADORES

El procesador: Camino de datos y control

HOJA DE PROBLEMAS N° 3

5.- Mostrar lo que se debe añadir a la Máquina de Estados Finitos de la figura, para implementar la Instrucción **jal**:



6.- Una Máquina Multiciclo, tipo MIPS, podría correr a 750 MHerz. La lentitud de los accesos a la Memoria de datos, con las Instrucciones **lw** y **sw**, hace que solo pueda correr a 500 MHerz. Si los ciclos de acceso a Memoria de Datos se descomponen en dos, podría ir a la frecuencia máxima de 750 MHerz.

Suponiendo el Programa compilador **gcc** (usar la tabla de frecuencias de instrucciones vista en “Aritmética para computadores”); ¿Cuántas veces es más rápida la máquina con accesos a Memoria de dos ciclos (750 MHz.) respecto a la otra? Suponer que Saltos y Bifurcaciones necesitan los mismos ciclos y que las Instrucciones de Inicialización y Aritméticas Immediatas se implementan como tipo R.

ARQUITECTURAS AVANZADAS DE PROCESADORES

El procesador: Camino de datos y control

HOJA DE PROBLEMAS N° 4

7.- Suponer tres Máquinas tipo MIPS (con el repertorio de Instrucciones dadas) con las siguientes características adicionales:

M1.- Camino de Datos Multiciclo con un Reloj de 50 MHz.

M2.- Igual que M1, solo que las actualizaciones de los Registros se hacen en el mismo ciclo de reloj que las operaciones de la ALU o la lectura de Memoria. En este caso el reloj es de 40 MHz.

M3.- Como la M2, pero los cálculos de las Direcciones Efectivas se hacen en el mismo ciclo de reloj como un acceso a Memoria. El reloj en este caso es de 25 MHz.

Encontrar la Máquina más rápida, usando los datos del compilador gcc (ver tabla de frecuencias).

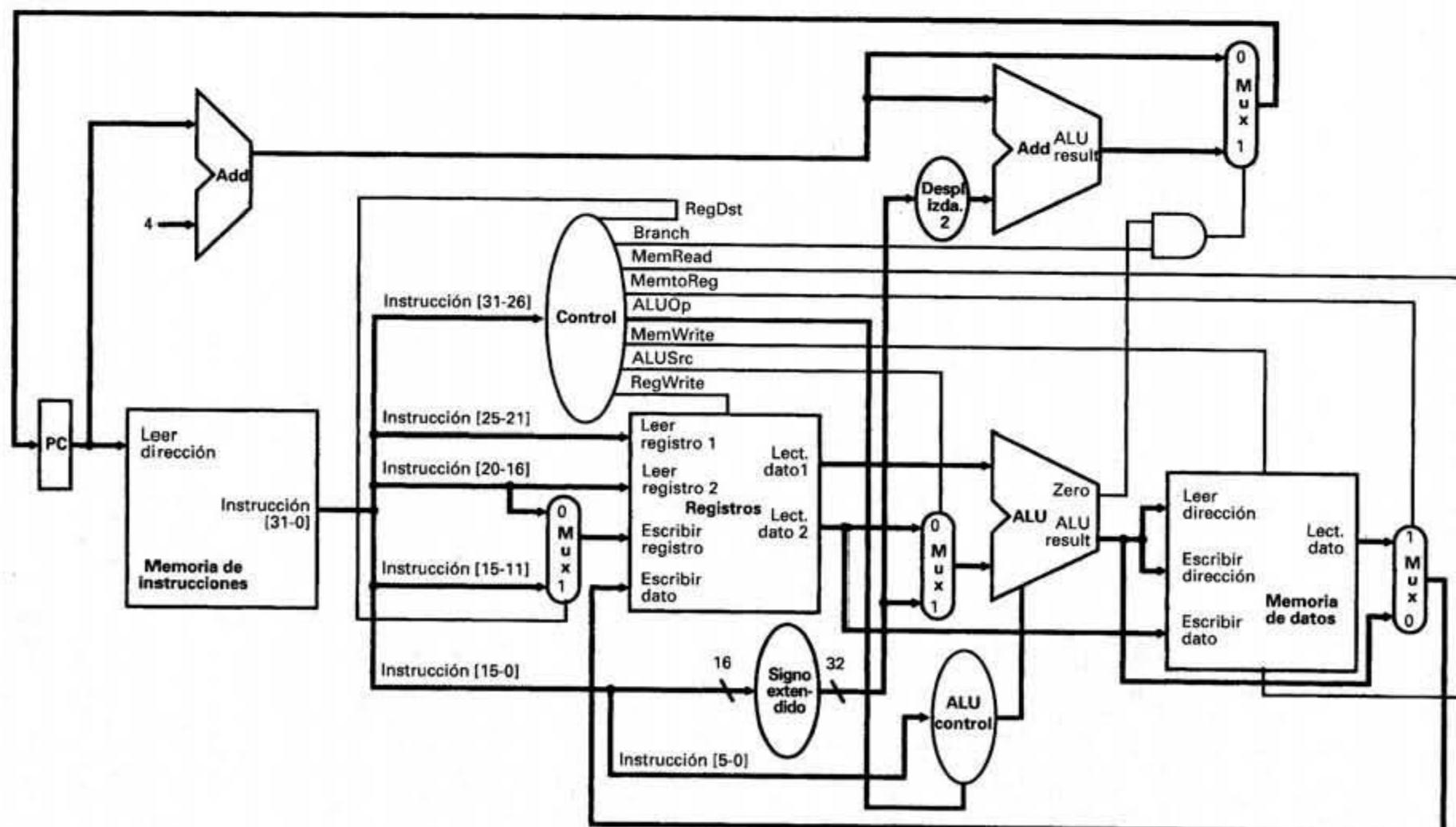
8.- Suponer que existiese una Instrucción **bcp**, que copia un bloque de palabras de Memoria a Memoria. Suponer que la dirección de inicio fuente esta en \$1 y la destino en \$2, y el número de palabras a copiar en el \$3 (≥ 0). Suponer también que los valores de estos Registros, así como el \$4, pueden ser destruidos al ejecutar esta instrucción (para que los registros puedan utilizarse de forma temporal durante la ejecución de la Instrucción)

Escribir el programa MIPS para implementar la copia del bloque.

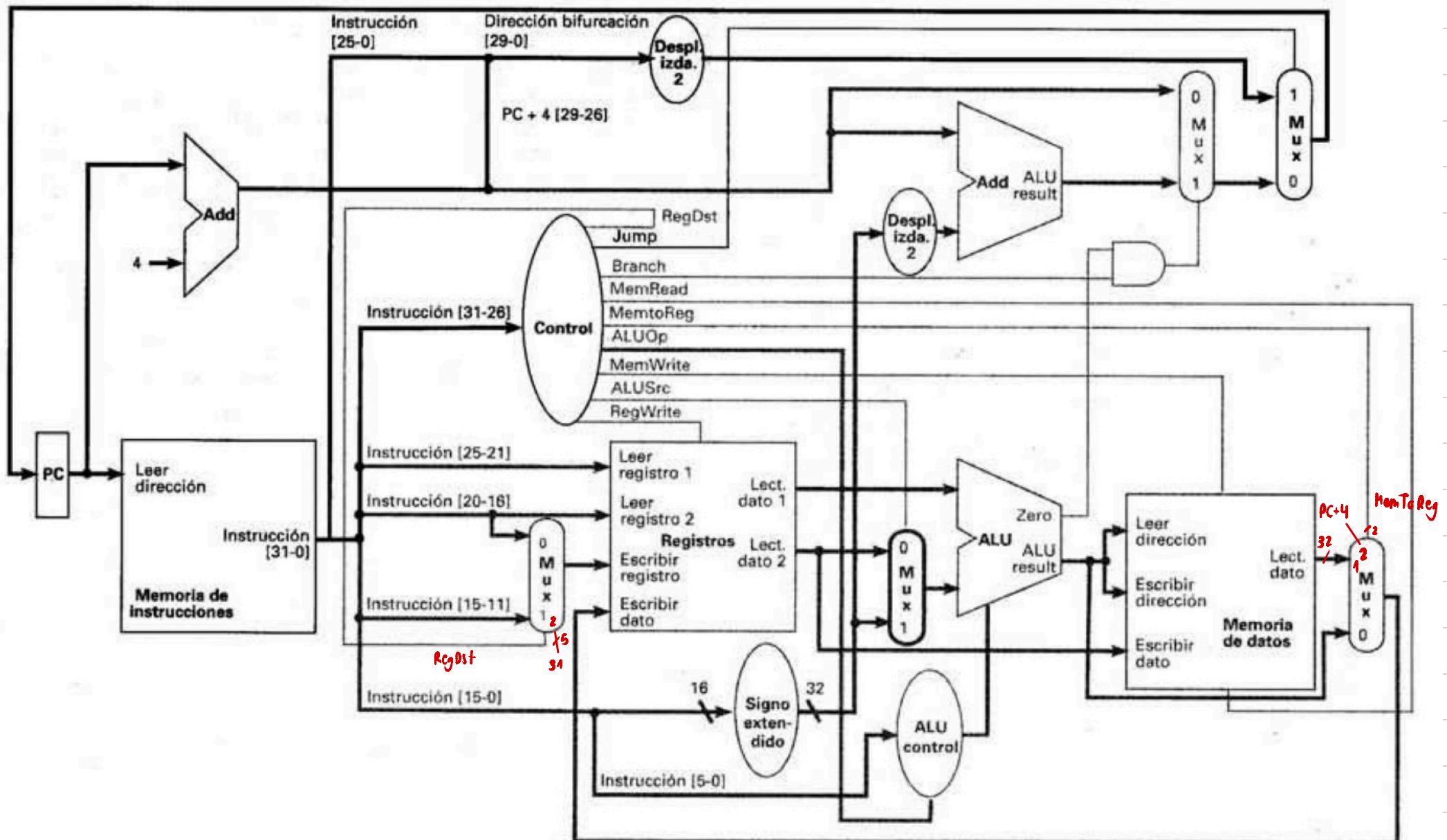
Suponiendo la copia de 100 palabras, ¿cuántas instrucciones se ejecutarán?

Usando la máquina multiciclo para los CPI, ¿cuántos ciclos son necesarios?

9.- Deseamos añadir la Instrucción **addiu** (suma inmediata sin signo) al camino de datos de un solo ciclo. Añadir los caminos de datos y señales de control necesarias a la figura:



1.- Camino de Datos y señales de Control necesarias para incorporar la Instrucción **jal** (jump and link) para el Camino de Datos Monociclo (las modificaciones se pueden realizar en la figura siguiente):



jal dir

$$r2 = PC + 4$$

$$r2 = 31 \equiv 11111$$

$$dir \in [25 \dots 0]$$

$$dir_PC = dir \oplus PC + 4 \oplus PC + 4 [31 \dots 28]$$

jr \$r2

MemTo Reg 1	MemTo Reg 0	Camino Dato
0	0	ALU resultado
0	1	Ler dato
1	0	PC + 4
1	1	PC + 4 (control)

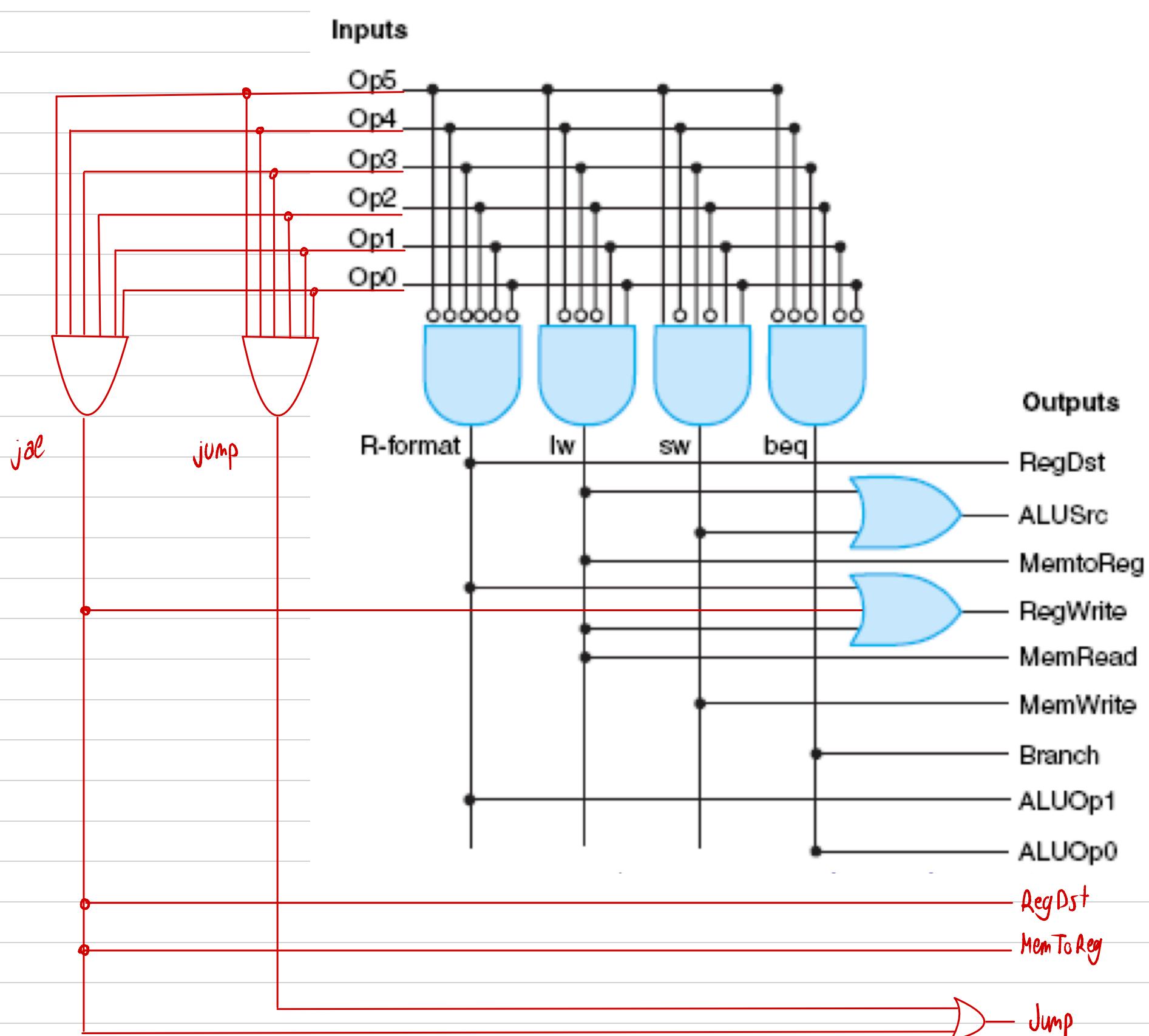
RegDest 1	RegDest 0	Camino Dato
0	0	Instrucción [20 ... 16]
0	1	Instrucción [15 ... 11]
1	0	11111 (31)
1	1	11111 (31)

2.- Ampliar la tabla de inicialización de las líneas de control para ver los valores que deben de presentar todas las líneas de Control que se añadieron en el ejercicio anterior para la Instrucción **jal**:

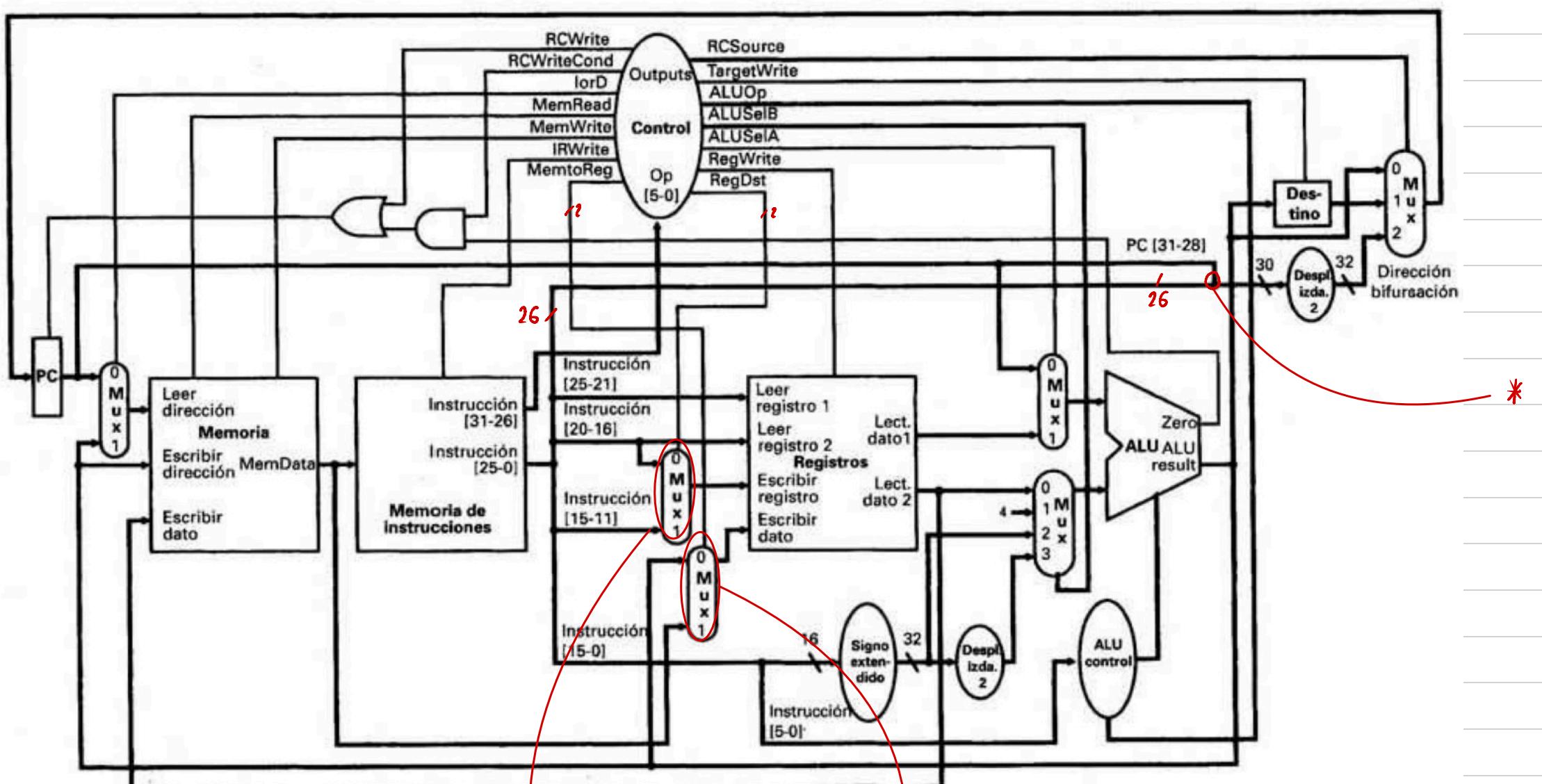
Instrucción	RegDst	ALUSrc	MemtoReg	RegWrite	MemRead	MemWrite	Branch	ALUOp1	ALUOp0	JUMP
Formato R	1	0	0	1	0	0	0	1	0	0
lw	0	1	1	1	1	0	0	0	0	0
sw	X	1	X	0	0	1	0	0	0	0
beq	X	0	X	0	0	0	1	0	1	0
jal	XX	X	XX	0	0	0	0	X	X	1
jal	1X	X	1X	1	0	0	0	X	X	1

Tabla de verdad de la unidad de control para los formatos R, lw, sw y beq

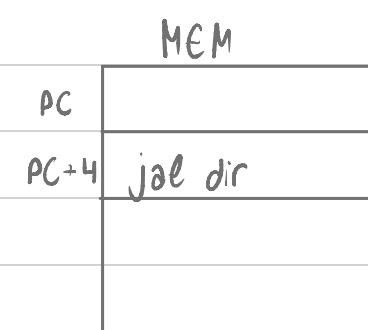
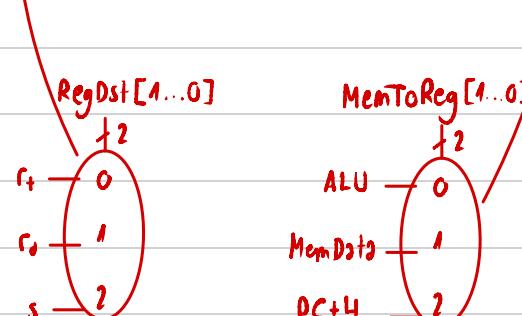
Control	Signal name	R-format	lw	sw	beq	jump	jal
Inputs	Op5	0	1	1	0	1	1
	Op4	0	0	0	0	1	1
	Op3	0	0	1	0	1	1
	Op2	0	0	0	1	1	1
	Op1	0	1	1	0	1	1
	Op0	0	1	1	0	1	1
Outputs	RegDst	1	0	X	X	XX	10
	ALUSrc	0	1	1	0	X	X
	MemtoReg	0	1	X	X	XX	10
	RegWrite	1	1	0	0	0	1
	MemRead	0	1	0	0	0	0
	MemWrite	0	0	1	0	0	0
	Branch	0	0	0	1	0	0
	ALUOp1	1	0	0	0	X	X
	ALUOp0	0	0	0	1	X	X
	Jump	0	0	0	0	1	1



3.- ¿Qué se debe añadir al Camino de Datos y a las Líneas de Control, para que se pueda implementar la Instrucción **jal** en el Camino de Datos Multiciclo, de forma que minimice el número de ciclos de reloj para su ejecución?



jal
 dir [25..0]
 $PC + 4 \rightarrow \$ra$
 $r_a = 31$
 $*PC + 4[3..0] \oplus dir[25..0] \oplus 00*$



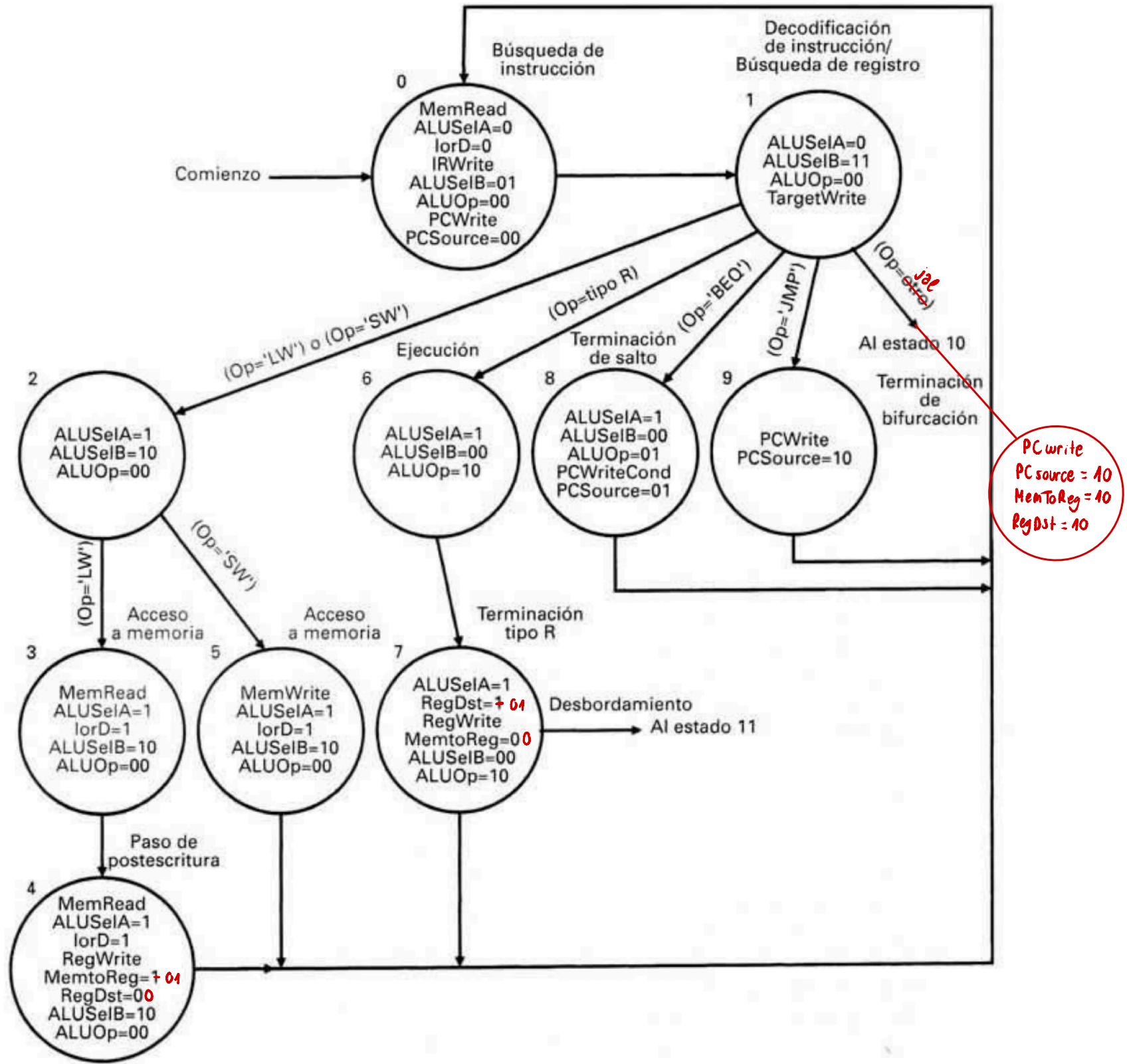
MemToReg1	MemToReg0	Escribir Dato
0	0	ALU
0	1	MemData
1	x	PC+4

RegDest 1	RegDest 0	Escribir Registro
0	0	r_t
0	1	r_d
1	x	11111(31)

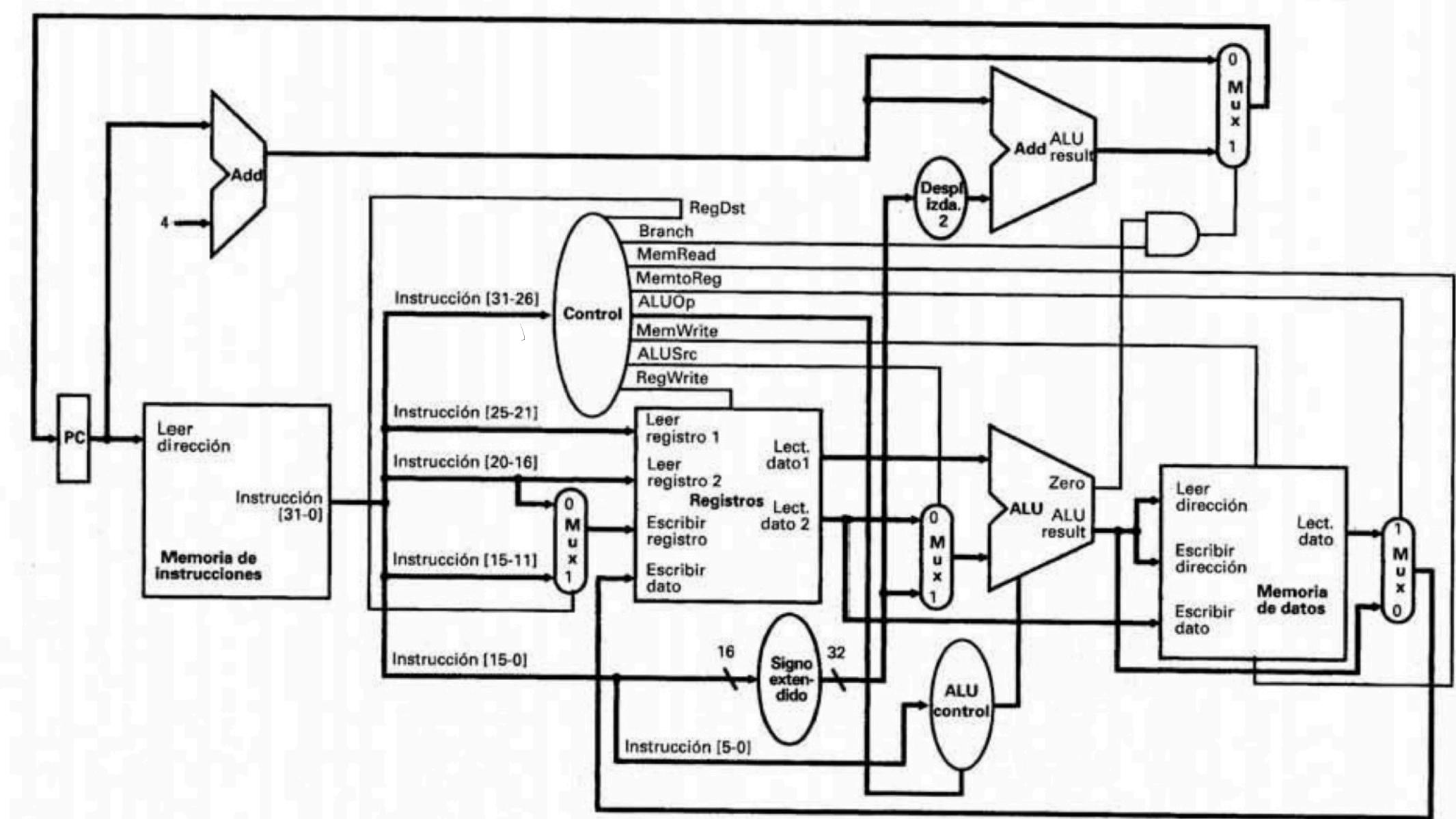
4.- Mostrar los pasos en la ejecución de la Instrucción **jal**, en el Camino de Datos Multiciclo, utilizando la misma descomposición de pasos que se muestra en la figura:

Nombre de paso	Acción para instrucciones tipo R	Acción para instrucciones de referencia a memoria	Acción para saltar	JUMP	Jal
Búsqueda de instrucción		IR=Memoria(PC) PC=PC+4;			
Decodificación de instrucción/Búsqueda de registros		A=Registro[IR[25-21]] A=Registro[IR[20-16]] Target=PC+signo extendido (IR[15-0])<<2			
Ejecución, cálculo de dirección o terminación de salto	ALUoutput=A op B	ALUoutput=A+signo extendido (IR[15-0])	si (A==B) entonces PC=Target	$PC = PC + 4 \oplus dir[25..0] \oplus 00$	\oplus $PC + 4 \rightarrow \$ra$
Acceso a memoria o terminación tipo R	Reg[IR[15-11]]=ALUoutput	dato-memoria=Memoria[ALUoutput] o Memoria [ALUoutput]=B			
Postescritura		Reg[IR[20-16]]=dato-memoria			

5.- Mostrar lo que se debe añadir a la Máquina de Estados Finitos de la figura, para implementar la Instrucción jal:



9.- Deseamos añadir la Instrucción **addiu** (suma inmediata sin signo) al camino de datos de un solo ciclo. Añadir los caminos de datos y señales de control necesarias a la figura:



addiu

suma inmediata

sin signo