			0	1	2	3	4	5	6	7	8	9	Α	В	С	D	E	F
С	0x00	0x00000000	0x00700293	0x0000004	0x00100313	0x0000008	0x0062f333	0x000000c	0x00030463	0x00000010	0xfff28293	0x00000014	0x0012d293					T
SP.		0x00000010																
		PC	SP	IR(hexa)		Instrucc	ción	PC		REGISTRO	VALOR		REGISTRO	VALOR				
	1	0x00000000		00700293		addi x5, x		0x00000004		x0 / zero			x18 / s2					
		Ejecución			ro x5 la suma e		x7 (7). No modifica			x1 / ra		_	x19 / s3					
	2	0x00000004		00100313		addi x6, x		0x00000008		x2 / sp			x20 / s4					
		Ejecución			ro x6 la suma e		x1 (1). No modifica			x3 / gp		_	x21 / s5					
	3	0x00000008		0062f333	and x6, x5, x6			0x0000000c		x4 / tp			x22 / s6					
		Ejecución			ores de x5 y x6		es escrito en x6. No			x5 / t0	0x7, 0x6, 0x3		x23 / s7					
	4	0x0000000c		00030463		beq x6, x0	O, 0x8	0x00000010		x6 / t1	0x1		x24 / s8					
			Si el va	lor de x6 es iqua	e x6 es igual a 0, realiza un salto de 2 instrucciones (PC + 0x0													
		Ejecución		(Como es falso,	continua en 0x	00000010 `			x7 / t2			x25 / s9					
	5	0x00000010		fff28293		addi x5, x	< 5, -1	0x00000014		x8 / s0 / fp			x26 / s10					
		Ejecución			jistro x5 la sum	a entre x5 (7) y	/ -1. No modifica me	emoria		x9 / s1			x27 / s11					
	6	0x00000014		0012d293		srli x5, x	5, 1	0x00000018		x10 / a0			x28 / t3					
		Ejecución	Realiza	a un Shift Right	al valor guardado en x5 (Una sola vez). No modifi			ica memoria		x11 / a1			x29 / t4					
	7									x12 / a2			x30 / t5					
		Ejecución								x13 / a3			x31 / t6					
	8									x14 / a4								
		Ejecución								x15 / a5								
	9									x16 / a6								
		Ejecución								x17 / a7								
	10																	
		Ejecución		•	•			•										
	11																	
		Ejecución				•												
	12																	
		Ejecución		•				•										
	13																	
		Ejecución																
	14																	
		Ejecución		•	•													
	15																	
		Ejecución		•	•			'										

			Memoria										
DC	0x00		0x0000000	0x00400593									
PC SP	UXUU		_	0x00400593									
5P			0x00000004 0x00000008										
			0x000000000000000000000000000000000000	0x00400693 0x0006a683									
				0x0006a683									
			0x00000010 0x00000014	0x0006a683									
			0x00000014	0x0080006f									
			_										
			0x0000001c 0x00000020	0x0000006f 0xfffa6737									
			0x00000020	0x111a6737									
			0x00000024		0x00000004	0x00000000							
			0x00000028	0x00c70633		000000000							
			0x0000002C										
			0x00000030	uxiediluoi									
			PC	SP	IR(hexa)		Instruccio	ián	PC	REGISTRO	VALOR	REGISTRO	VALOR
		4		55	00400593		addi x11. x		0x000000c		VALOR		VALUR
		1	0x00000008	-		v11 lo s::=== : :		-,		x0 / zero x1 / ra		x18 / s2 x19 / s3	
		2	Ejecución 0x0000000c	-	0005a603	i x i i ia suma en		4 (4). No modifica m	0x0000010	x1 / ra x2 / sp		x20 / s4	
		2	_			5a603 lw x12 0 x11 Escribe en x12 lo que vale en la posicion de memoria 0(x11)						x20 / \$4 x21 / \$5	
		2							0x00000014	x3 / gp		X21 / S5	
		3		_		1, 1, 1,							
		-	Ejecución	E		o x13 la suma entre x0 (0) y 0x4 (4). No modifica memoria lw x13 0 x13 0x00000018				4.00		20.4.2	
		4	0x00000014		0006a683	12			0x00000018	x4 / tp		x22 / s6	
		5	Ejecución			13 io que vaie e		de memoria 0(x13)	0.0000000	x5 / t0		x23 / s7	
		5	0x00000018		fed606e3		beq x12 x1		0x00000020	x6 / t1		x24 / s8	
			F	Si e	Si el valor de x12 es igual al de x13, retrocede 5 instrucciones (PC - 0 Como es falso, continua en 0x00000020				000020).	7.40		05 / 0	
		-	Ejecución			omo es falso, co			10.0000000	x7 / t2		x25 / s9	
		6	0x00000020		0080006f		jal x0 8		0x00000028	x8 / s0 / fp		x26 / s10	
				Alm		al del PC en x0, ¡ ue valiendo 0 y a		rlo en x0 este no se	modifica				
			Ejecución			ue vallendo o y a zando 2 instrucci				x9 / s1		x27 / s11	
		7	0x00000028		fffa6737	Lando Z mando	lui x14 0xf		0x0000002c	x10 / a0		x28 / t3	
		,	Ejecución			cribe en el regis			000000020	x11 / a1	0x00000004	x29 / t4	
		8	0x0000002c		9fd70713	l l l l l l l l l l l l l l l l l l l	addi x14 x14		0x00000030	x12 / a2	0x0005a603 , 0x00000000	x30 / t5	
		0	Ejecución	Fect		I la suma entre		y -1539. No modifica		x13 / a3	0x0005a603 , 0x00000000	x31 / t6	
		9	0x00000030	LSCI	00c70633	14 la sulla citile	add x12 x14		0x00000034	x14 / a4	0xfffa6000, 0xfffa59fd	X317 to	
		3		Fassiba as		Ima entre el valor de la posiciones x14 y x12. No modifica memoria				x15 / a5	Oxinadodo, Oxinadora		
		10	Ejecución	Escribe en	02b62423	ma entre ei vaio			0x0000038				
		10	0x00000034				sw x11 40(· /	000000038	x16 / a6			
		11	Ejecución 0x00000038		fedff06f	posicion de mer		alor del registro x12	0x00000018	x17 / a7			
		111	0x00000036	Δlm		jal x0 -20 0x00000018 al del PC en x0, pero al guardarlo en x0 este no se modifica							
						ue valiendo 0 y a			modifica				
			Ejecución		retroce	diendo 5 instruc	ciones (PC - 0)	x000000020)					
		12	0x00000018		0000006f		jal x0 0	0	0x00000018				
				Alma				ırlo en x0 este no se	modifica				
						ue valiendo 0 y a							
			Ejecución			zando 0 instrucci							
		13	0x00000018		0000006f	1.1.1.00	jal x0 0		0x00000018				
				Alma				rlo en x0 este no se	modifica				
			Ejecución		y sigue valiendo 0 y actualiza el valor del PC, avanzando 0 instrucciones (PC + 0x00000000) 0000006f jal x0 0								
		14	0x00000018										
		1.7	5,0000010	Alma		al del PC en x0, pero al guardarlo en x0 este no se modifica							
					y sig	ue valiendo 0 y a	actualiza el val	lor del PC,					
			Ejecución			ando 0 instrucci							
		15											
			Ejecución										
		15											
			Ejecución										

2. a) Indicar en qué posiciones de memoria se encuentra cada etiqueta.

fin: 0x00000014resta: 0x00000018sigo: 0x00000020

b) Indicar el desplazamiento de las llamadas a etiquetas.

Tanto *j resta* en la línea 3 como en la 7 retroceden dos instrucciones (PC - 0x00000008)

c) Indique el rango de constantes, en decimal y binario que pueden utilizarse en la instrucción *li*. ¿Coinciden con el rango del imm de la instrucción ADDI?

El rango de constantes de li en decimal es de [-(2^31); (2^31)-1] y en binario [10 ... 00; 01...11]. Y no coinciden con el rango del imm de ADDI ya que son de 12 bits.

d) ¿Cómo resuelve los valores inmediatos que no son representables en 12 bits C2?

Cuando li recibe un valor imm no representable en 12 bits c2 se dividen los 32 bits en los primeros 20 para un lui y los 12 restantes para un addi (Si los 12 bits del addi son 0, no se hace el addi).

e) ¿Cuál es el valor final de a1?

0x2114

f) ¿Cuál es el valor final de PC?

0x00000014 <fin> por el loop

g) Listar la secuencia descripta por el pc

- **1.** 0x00000000
- **2.** 0x00000004
- **3.** 0x00000008
- **4.** 0x0000000c
- **5.** 0x00000010
- **6.** 0x00000014
- 7. 0x00000018
- **8.** 0x0000001c

- **9.** 0×000000020 (a0 = 2114)
- **10.** 0x00000018
- **11.** 0x00000014
- **12.** 0x0000001c
- **13.** 0x00000020 (a0 = 0)
- **14.** 0x00000018
- **15.** 0x00000014
- **16.** 0x00000014
- **17.** (loop)
- h) Reemplazar la segunda instrucción *li a1,2114* de modo que *a1* sea a0 dividido 2 con una única instrucción.

srli a1, a0, 1

3. b) Suponga que el programa hubiese sido cargado en la posición 0x0000 y el PC comienza con ese valor. ¿Cambia la ejecución del programa? ¿De qué manera? ¿Por qué?

La ejecución del programa no cambia, ya que los saltos de instrucción siguen estando en las etiquetas correctas y los cambios en memoria son con valores de los mismos registros.