

[illegible]



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

- fin: 0x00000014
- resta: 0x00000018
- sigo: 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  $[10 \dots 00; 01 \dots 11]$ . 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 descrita por el pc**

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

9. 0x00000020 (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.