FUNDAMENTOS DE COMPUTADORES
5 de septiembre de 2013. Examen parcial del 2º cuatrimestre.

Nombre_____DNI_____

Apellidos_____Grupo_____

Ejercicio 1 (3 puntos). Responde a las cuestiones suponiendo que el vector V está almacenado a partir de la dirección de memoria 0x0C000000, que el código se encuentra a continuación de los datos y que las pseudo-instrucciones ocupan el mismo espacio que las instrucciones.

```
# Codigo 1
.global start

.data
V: .word 12,21,13,14,5,9
N: .word 6

.bss
CuentaTotal: .space 4

.text
start:      ldr R0,=V
            ldr R2,=N
            ldr R1,[R2]
            mov R2,#0
            mov R3,#0
bucle:      cmp R2,R1
            beq fin_bucle
            ldr R4,[R0]
            and R4,R4,#1
            add R3,R3,R4
            add R2,R2,#1
            add R0,R0,#4
            b bucle
fin_bucle:  ldr R1,=CuentaTotal
            str R3, [R1]
            b .

.end
```

```
#Código 2
.global start

.data
V: .word 12,21,13,14,5,9
N: .word 6

.bss
CuentaTotal: .space 4

.text
start:      mov sp,#0x0C200000
            ldr R0,=V
            ldr R2,=N
            ldr R1,[R2]
            bl Cuenta
            ldr R2,=CuentaTotal
            str R0,[R2]
            b .

Cuenta:     PRÓLOGO_1
            mov R4,#0
            mov R5,#0
            mov R6,R0
            mov R7,R1
bucle:      cmp R4,R7
            beq fin_bucle
            ldr R0, [R6]
            bl Comprobar
            add R5,R5,R0
            add R4,R4,#1
            add R6,R6,#4
            b bucle
fin_bucle:  mov R0,R5
            EPÍLOGO_1
            mov pc,lr

Comprobar:  PRÓLOGO_2
            mov R4,#1
            and R0,R0,R4
            EPÍLOGO_2
            mov pc,lr

.end
```

a) (0,75 puntos) ¿En qué dirección de memoria del código 1 está almacenada la variable N? ¿Y la primera instrucción del bucle para el código 1? Razona las respuestas.

b) (0,75 puntos) ¿Cuál es el contenido final de la variable de memoria CuentaTotal en el código 1? ¿Y de los registros R0, R1 y R2? Razona la respuesta.

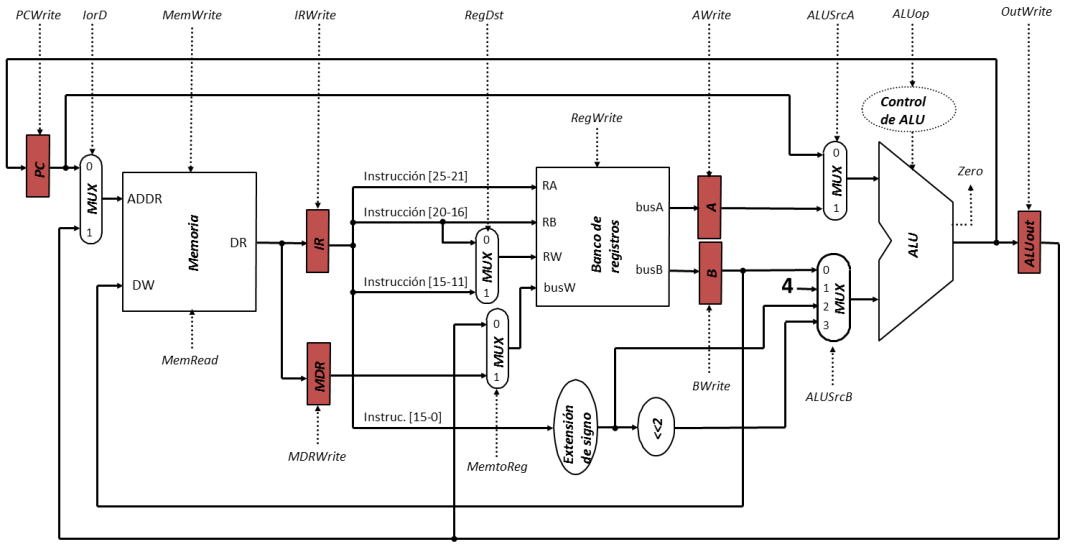
c) (1.5 puntos) Supongamos que queremos estructurar el código con subrutinas, y lo rescribimos en código 2. Completa el prólogo y epílogo de cada una de las dos subrutinas, explicando por qué incluyes cada instrucción.

Ejercicio 2 (3 puntos). Se desea añadir al procesador multiciclo la instrucción JALR Rd, Rs, con el formato tipo R pero código de operación 001001:

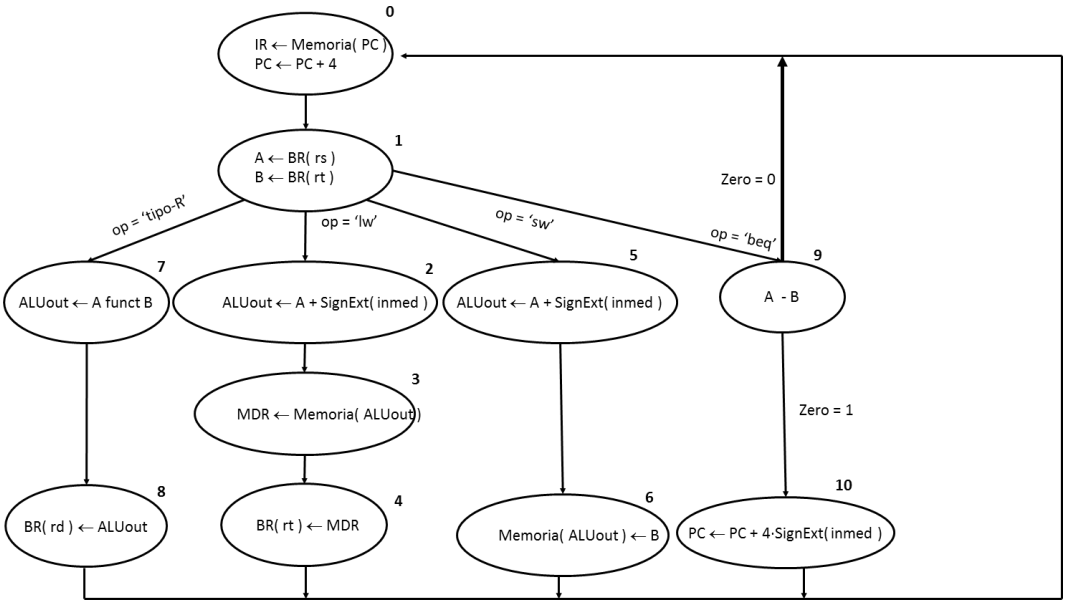
JALR Rd, Rs # Temp <- BR[Rs], BR[Rd] <- PC+8 , PC <- Temp

Nótese que Temp no es un registro físico, sino que se introduce en la descripción para aclarar qué ocurre en el caso de que Rs=Rd: en PC se escribe el valor antiguo de Rs, antes de que este mismo registro se actualice con el valor PC+8.

a) (1 punto) Indica todos los cambios que tendrían que realizarse a la ruta de datos para poder ejecutar esta instrucción.



b) (1 punto) Describir los cambios necesarios en la máquina de estados del controlador para poder ejecutar correctamente esta instrucción.



c) (1 punto) Indica los valores que van tomando todos los registros implicados en la ejecución de la instrucción JALR R1,R3, sabiendo que los campos Rt, SHAMT y FUNCT se dejan a 0. El estado actual de la máquina es:
   - R0=0x000000FA
   - R1=0x00000030
   - R2=0x0C001600
   - R3=0x0C00E040
   - PC=0x0C000020

Ejercicio 3 (3 puntos). Supongamos una jerarquía de memoria que consta de una Mp direccionable por palabras, donde cada palabra tiene una anchura de 32 bits, y una Mc de emplazamiento directo con 128 bloques, siendo cada bloque de 512 palabras. Además sabemos que:

- Las direcciones de Mc tienen un campo de etiqueta de 8 bits.
- El tiempo de acceso a la Mc es 1 ciclo de reloj.
- La penalización por fallo es 130 ciclos de reloj.
- El tiempo de acceso a Mp es 10 ciclos de reloj.

Sobre esta jerarquía se ejecuta 10 veces seguidas una cadena de referencias entre las direcciones 0xF59600 y 0xF698C7, ambas inclusive, estando la cache inicialmente vacía.

a) (0,5 puntos) ¿Cuál es el tamaño de la Mp expresado en megabytes?
b) (1 punto) Determina el número total de fallos producidos.
c) (0,5 puntos) Teniendo en cuenta la siguiente definición, determina la *Tasa de Fallos* en este problema.

$$Tasa\ de\ Fallos = \frac{Número\ total\ de\ fallos}{Número\ total\ de\ referencias\ a\ memoria}$$

d) (0,5 puntos) Determina el tiempo medio de acceso a memoria.
e) (0,5 puntos) Calcula la mejora de velocidad que se obtiene al emplear esta jerarquía de memoria, en comparación con la alternativa de usar solamente la Mp.

Ejercicio 4 (1 punto). Dado un programa en lenguaje ensamblador del ARM que comienza de la siguiente manera:

```
.global start
.data
luces:    .word 0
.bss
teclas:  .space 4
.text
start:   … … …
         … … …
```

Escriba en lenguaje ensamblador del ARM la secuencia de instrucciones que permite:

a) (0,33 puntos) Escribir “11” en los bits 9 y 10 de la variable “luces” dejando los demás bits inalterados.

b) (0,33 puntos) Invertir los bits 6 y 7 de la variable “luces” dejando los demás bits inalterados.

c) (0,33 puntos) Si el bit 9 de la variable “teclas” es un 1, saltar a la instrucción que tiene como etiqueta “apagado”. En caso contrario seguir la ejecución secuencial.

Nota: Cada apartado es independiente de los demás. Se considera que el bit menos significativo de una palabra es el bit 0.