UNIVERSITAT D'ALACANT
UNIVERSIDAD DE ALICANTE
DPNT DE TECNOLOGIA INFORMÀTICA I COMPUTACIÓ
DPTO DE TECNOLOGÍA INFORMÁTICA Y COMPUTACIÓN

# Estructuras de Computadores – (34010)

Examen (3 de Junio de 2019)

**Pregunta 1.** **(1.5 puntos)**

Se desea multiplicar mediante el algoritmo de Booth los siguientes números de cuatro bits, Multiplicando=1011 y Multiplicador=0100. Rellenar la tabla siguiente con el proceso de multiplicación detallando cada uno de los pasos que ocurren.

**Pregunta 2.** **(1 punto)**

a) ¿Cómo se utiliza los registros que tiene el controlador de DMA? (0.5 puntos).
b) Indica y describe las señales que utiliza el controlador de DMA para transferir información a la memoria. (0.5 puntos).

**Pregunta 3.** **(2.5 puntos)**

Un computador posee una CPU de 20 bits de bus de direcciones y de 16 bits de longitud de palabra y señal de lectura/escritura: R/$\overline{W}$. Se desea conectar esta CPU a una memoria con las siguientes características:

- 256K x 16 de memoria ROM comenzando en la dirección 20000 H.
- 256K x 16 de memoria RAM comenzando en la dirección C0000 H.

Se dispone de los siguientes tipos de chips de memoria para elegir:

| RAM | ROM |
|---|---|
| 64K x 16<br>128K x 8<br>256K x 8<br>256Kx1 | 64K x 8<br>128K x 16<br>256K x 1 |
| Selección chip RAM: CS<br>Señal de Lectura RAM: $\overline{OE}$<br>Señal de Escritura: $\overline{WE}$ | Selección chip ROM: $\overline{CS}$<br>Señal de Lectura ROM: OE |

Se pide:

a) Diseñar el mapa de memoria descrito utilizando el **menor número** de chips posible de entre los tipos que aparecen en el cuadro anterior e **indicando claramente** las **direcciones** de **comienzo** y **final** de cada chip, así como señalando en el mapa cuáles son **los bits de selección** y cuáles **los de direccionamiento**. (1,25 ptos.)

b) Realizar el diagrama de conexiones completo entre la CPU y la memoria según las especificaciones del enunciado, utilizando para la **selección** de chips **un solo decodificador** y las puertas logicas que se necesiten (1,25 ptos.)

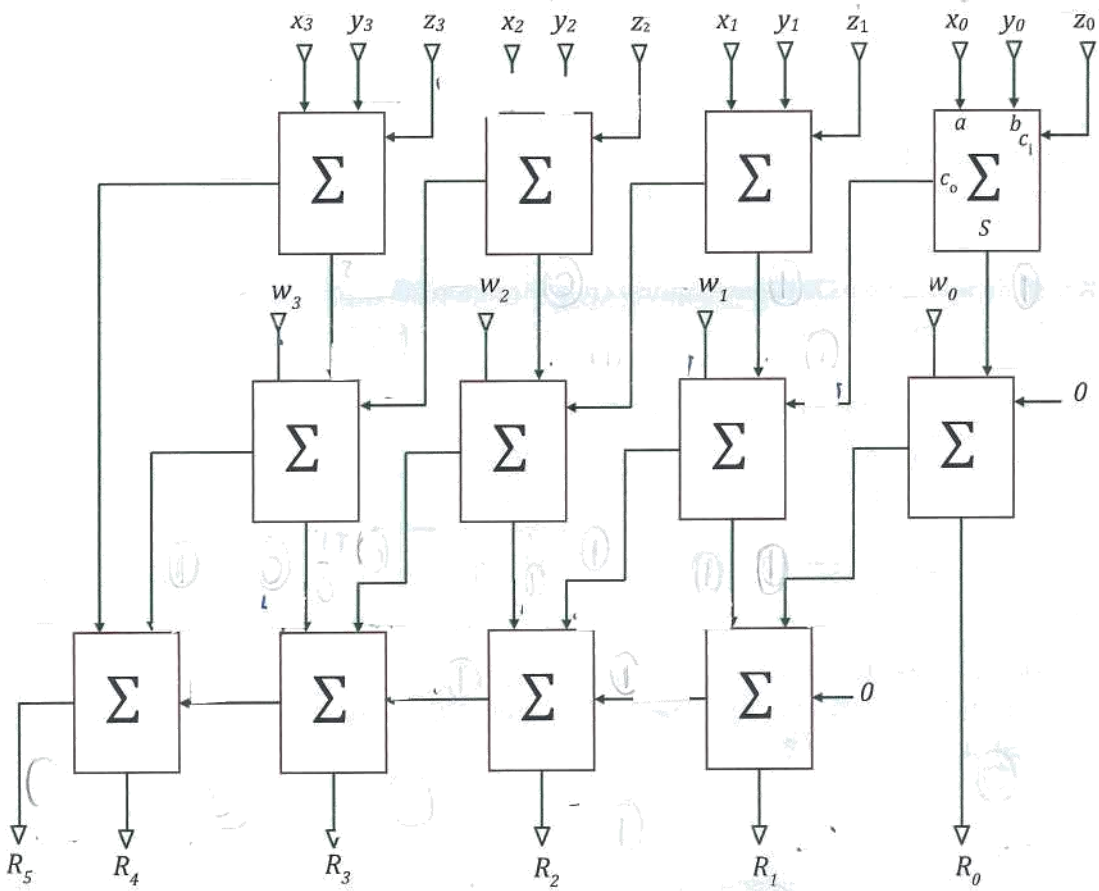
**Pregunta 4.** **(2.5 puntos)**

El esquema de la figura corresponde a un circuito sumador capaz de sumar 4 números ($W$, $X$, $Y$, $Z$) de 4 bits ($n_3$, $n_2$, $n_1$, $n_0$) de forma eficiente. Los bloques sumadores están construidos con semisumadores en los que las puertas que los implementan tienen todas un retardo de 1T

a) Si ($w_3$, $w_2$, $w_1$, $w_0$) =($x_3$, $x_2$, $x_1$, $x_0$) = ($y_3$, $y_2$, $y_1$, $y_0$)= ($z_3$, $z_2$, $z_1$, $z_0$) = 1011, realiza la suma sobre el propio esquema hasta obtener ($R_5$, $R_4$, $R_3$, $R_2$, $R_1$, $R_0$) e indica el tiempo que se tarda en obtener cada uno de ellos.

b) Indica de forma razonada cuál sería tiempo de respuesta del circuito.

c) Los cuatro sumadores que constituyen el nivel inferior tienen una estructura de sumador con acarreo propagado. Si los sustituimos por uno con acarreo anticipado ¿cuál sería el nuevo tiempo de respuesta?



**Pregunta 5.** **(2.5 puntos)**

**5.1** (0,7 puntos) Se quiere ejecutar la siguiente instrucción con formato tipo I y código de operación 0x2B en la ruta de datos monociclo: sw \$15, 4(\$13)

La posición de memoria $2000_{10}$ contiene dicha instrucción, es decir contiene el valor 0xADAF0004. Se quiere mostrar lo que ocurre durante su ejecución en la ruta de datos de la figura 1, rellena para ello las partes del diagrama señaladas con el símbolo (?) con los valores **en decimal** que contendrán. El contenido de los registros en decimal es el siguiente:

| Registros | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| \$0 | \$1 | \$2 | \$3 | \$12 | \$13 | \$15 | \$16 | \$20 | \$31 |
| 0 | 0 | 0 | 32 | 4 | 100 | 30 | 26 | -32 | -1 |

Indica además el valor de las señales de control en la ejecución de la instrucción.

**5.2.-** (1,8 puntos) En la figura 2 se muestra la ruta de datos multiciclo del MIPS a la que se han realizado unas pequeñas modificaciones para que se pueda ejecutar también una nueva instrucción, **Addm** que permite guardar en memoria la suma del contenido de un registro y el contenido de una posición de memoria. La instrucción sigue el formato tipo R y tiene la siguiente forma general:

Addm rd, rs, rt # M[rd] ← rs + M[rt]

a) Obtén las acciones a realizar en cada ciclo de reloj mediante lenguaje de transferencia de registros (por ejemplo: PC ← PC +4) y rellena la tabla de salida de la figura 3 con el valor de las señales de control en cada ciclo de ejecución de la nueva instrucción que no sea cero. (Tacha las columnas que no necesites l) . La nueva instrucción se debe ejecutar **con el menor** número posible de ciclos de reloj. (1 punto)

b) Para tener el mismo efecto que Addm, se necesita la ejecución consecutiva de las instrucciones: lw, add y sw. Para estas tres instrucciones y debido a los cambios en la nueva ruta de datos, indica en qué ciclos de reloj se verán afectadas las señales de control a activar y su nuevo valor. Si el ciclo de reloj es de 50Mhz, compara el tiempo de ejecución de la nueva instrucción Addm con el conjunto de las tres instrucciones con el mismo efecto. (0,8 puntos)

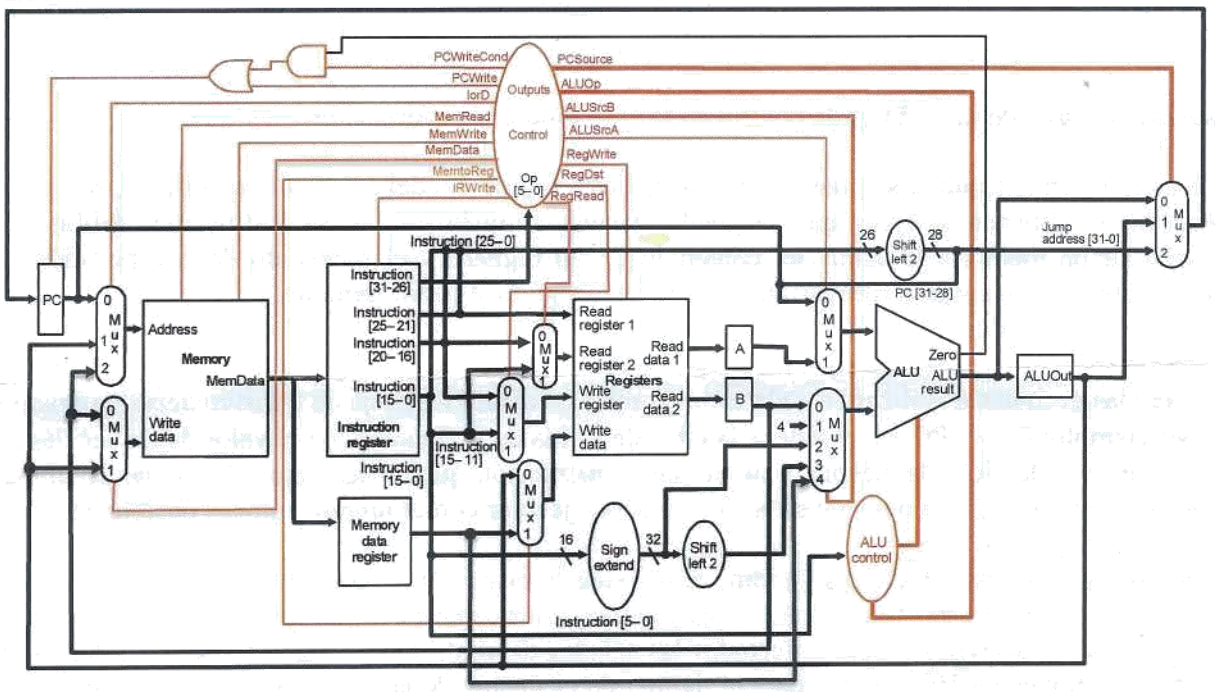


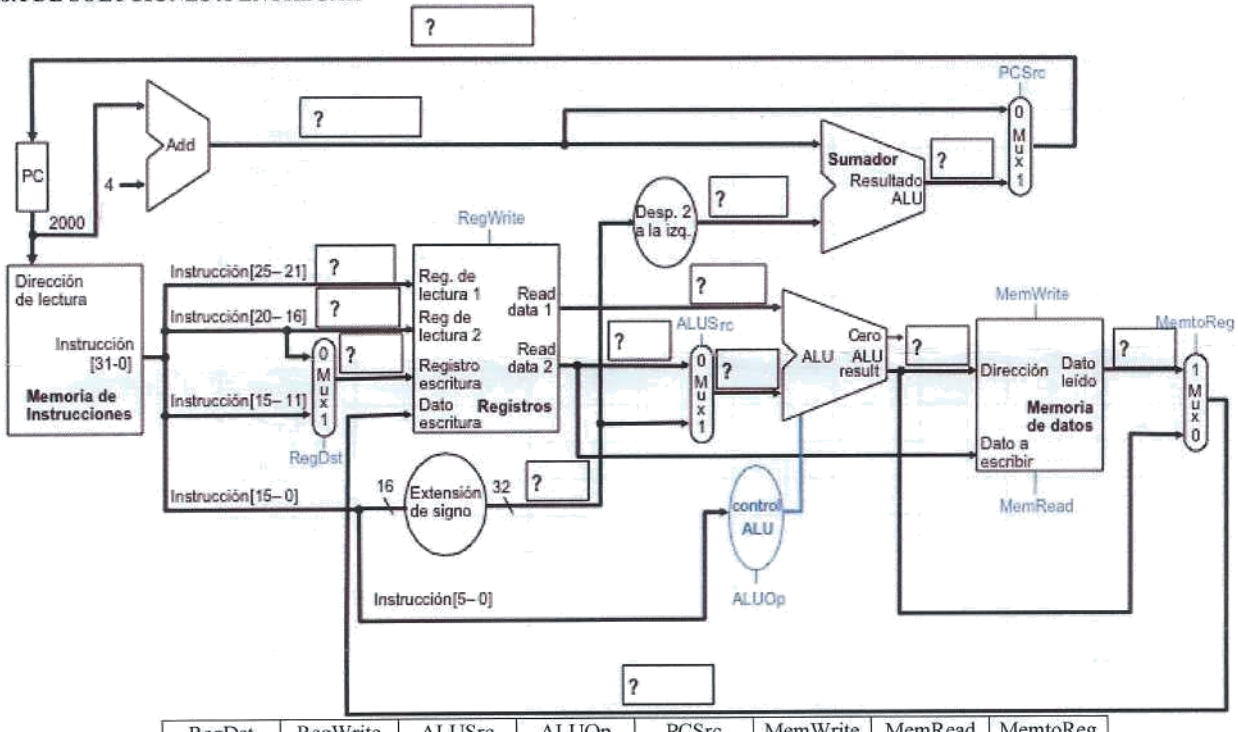

**Figura 2.** Ruta de datos multiciclo modificada

UNIVERSITAT D'ALACANT
UNIVERSIDAD DE ALICANTE
DPNT DE TECNOLOGIA INFORMÀTICA I COMPUTACIÓ
DPTO DE TECNOLOGÍA INFORMÁTICA Y COMPUTACIÓN

| | | | | | | |
|---|---|---|---|---|---|---|
| R | op | rs | rt | rd | shamt | funct |
| I | op | rs | rt | despl | | |
| J | op | direcc | | | | |

Formatos de instrucción

## HOJA DE SOLUCIONES A ENTREGAR




| RegDst | RegWrite | ALUSrc | ALUOp | PCSrc | MemWrite | MemRead | MemtoReg |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

**Figura 1.** Ruta de datos monociclo y tabla de señales de control a rellenar.

Ap. Correus 99 -:- E-03080 ALACANT -:- Tel. 96 / 590 36 81 -:- Fax 96 / 590 96 43

| Ciclo1 | Ciclo 2 | Ciclo 3 | Ciclo 4 | Ciclo 5 | Ciclo 6 | Ciclo 7 | Ciclo 8 | Ciclo 9 | **Señales de control** |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PCWriteCond |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PCWrite |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | IorD |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | MemRead |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | MemWrite |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | MemData |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | MemToReg |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | IRWrite |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PCSource |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ALUOp |
| 000 | 000 | 000 | 000 | 000 | 000 | 000 | 000 | 000 | ALUSrcB |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ALUSrcA |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RegWrite |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RegDst |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RegRead |

**Figura 3.** Tabla de salida para *Addm*