2002 Junio - 2ª semana

Test - Junio 2ª semana:

1: Una memoria caché por correspondencia directa utiliza 8 palabras/bloque y su capacidad total son 1K palabras. La memoria principal tiene una capacidad de 64K palabras. En un determinado instante, la dirección 6B59, expresada en hexadecimal, de la memoria principal se encuentra en la caché. Decir si las siguientes afirmaciones son ciertas:

I. Esta dirección se corresponde con el bloque 107, expresado en decimal, de la memoria caché.

II. Del enunciado puede deducirse que la dirección 675E, expresada en hexadecimal, no está en la caché.

A) I: sí, II: sí. B) I: sí, II: no. C) I: no, II: sí. D) I: no, II: no.

2: Sea un sistema de memoria caché con dos niveles, donde la memoria caché de nivel 1 es la más cercana a la CPU y la memoria de nivel 2 es la más cercana a la memoria principal. Indique si las siguientes afirmaciones son verdaderas:

I. En algún momento existirá una copia de todos los bloques de la caché de nivel 2 en la caché de nivel 1.

II. Utilizando una política de post-escritura, la memoria caché de nivel 2 tiene siempre una copia exacta de los bloques de la memoria caché de nivel 1.

A) I: sí, II: sí. B) I: sí, II: no. C) I: no, II: sí. D) I: no, II: no.

3: Un pequeño computador dispone de 16 líneas de direcciones $\wedge_{15}$-$\wedge_0$ Su unidad de memoria está compuesta tanto por módulos de memoria ROM como de RAM. La ROM está formada por un único módulo y ocupa las direcciones más bajas de la memoria, comenzando en la dirección 0. La RAM, con 48 Kbytes de capacidad, ocupa las direcciones restantes. Indique si las siguientes afirmaciones son verdaderas:

I. La función lógica de la señal de habilitación del módulo de memoria ROM es $CS_{ROM} = \wedge_{15}\wedge_{14}$

II. Los dos bits menos significativos de la dirección, A_1-A_0, se pueden utilizar para distinguir si una dirección de memoria corresponde a ROM o a RAM.

A) I: sí, II: sí. B) I: sí, II: no. C) I: no, II: sí. D) I: no, II: no.

4: Una memoria de acceso no aleatorio con velocidad de transferencia de 2×10^6 bits/seg, emplea en promedio 2 mseg en colocar en su posición la cabeza de lectura-escritura. ¿Cuál es el tiempo medio que tarda en leer o escribir 10^3 bytes?

A) 6 mseg B) 2 mseg C) 4 mseg D) Ninguna de las anteriores

5: Un sistema jerárquico de memoria tiene una memoria caché de 256 palabras, dividida en particiones de 8 palabras y con un tiempo de acceso de 20 nseg, y una memoria principal de 1024 Kpalabras con un tiempo de acceso de 200 nseg. Cuando se produce un fallo, primero se mueve el bloque completo a la memoria caché y después se lee el dato desde la caché. Si la tasa de acierto de la caché es del 90%, ¿cuál es el tiempo de acceso medio de este sistema?

A) 178 nseg B) 180 nseg C) 220 nseg D) Ninguna de las anteriores

6: Indique cuál de las siguientes afirmaciones sobre los buses es cierta:

A) Los buses transportan únicamente datos y direcciones.

B) Los buses pueden ser compartidos por dispositivos de muy distintas velocidades.

C) No es necesario un método de arbitraje para evitar que varias unidades vuelquen simultáneamente datos al bus.

D) Ninguna de las anteriores.

7: En un computador que usa E/S controlada por programa, el dispositivo de E/S tarda 6 mseg en tener disponible el dato solicitado. Suponiendo que el computador sólo se dedica a E/S, que el bucle de espera se implementa con una única instrucción, que la lectura de un dato y solicitud del siguiente dato por parte de la CPU lleva 10 instrucciones, y que cada instrucción de la CPU se ejecuta en 200 mseg, ¿qué tanto por ciento de su tiempo dedica la CPU al bucle de espera?

A) 0%. B) 99%. C) 75%. D) Ninguna de las anteriores

8: 8.- Sean dos números de 12 bits representados en código BCD: X=100100110101, Y=000101010001. El resto de la división entera X / Y, expresado en código binario, es:

A) 111000110000 B) 000000000110 C) 000000011101 D) Ninguno de los anteriores.

Solución

1: [Ver las páginas 81 y ss. del texto base de teoría.] La memoria principal de 64 Kpalabras necesita 16 bits para su direccionamiento ($2^{16} = 64K$). La memoria caché tiene 2^{10} (= 1K) palabras, luego hay (2^{10} palabras) / (2^3 palabras / bloque) = 2^7 bloques en la caché.

Al tratarse de correspondencia directa, una dirección de memoria principal se divide en tres campos: Etiqueta, Bloque y Palabra. El campo Bloque tendrá una longitud de 7 bits y el de Palabra, 3. Por tanto, el campo Etiqueta tendrá una longitud de 16 - (7 + 3) = 6 bits.

La dirección hexadecimal 6B59 se expresa en binario como 0110 1011 0101 1001, por lo que agrupando sus bits en los tres campos resulta Etiqueta = 011010, Bloque = 1101011 y Palabra = 001. Por tanto:

I. Bloque = $1101011_{(2} = {}^{107}{}_{(10}$

II. La dirección 675E se puede expresar como 011001 1101011 110. Obsérvese que el campo Bloque de ambas direcciones es igual, por lo que la correspondencia directa asigna a ambas direcciones el mismo

bloque de caché. Ahora bien, dado que sus campos Etiqueta son diferentes, es imposible que ambas palabras se encuentren simultáneamente en la caché. Como el enunciado afirma que la palabra de dirección 6B59 se encuentra en la caché, es imposible que también se encuentre la de dirección 675E.

Respuesta: A (I: sí, II: sí.)

Otro posible enfoque consiste en hacer uso de la expresión:

$$(\text{Bloque caché}) = (\text{Bloque } M_p) \bmod (\text{Número de bloques de la caché})$$

La memoria principal tiene (64 Kpalabras) / (8 palabras / bloque) = 8192 bloques (numerados de 0 a 8191). Una dirección de memoria principal se encuentra en el bloque de memoria principal:

$$(\text{Bloque } M_p) = (\text{Dirección } M_p) \setminus (\text{Número de palabras por bloque})$$

(el símbolo \ representa en este caso la división entera). Aplicando esta última expresión se llega a que:

6B59: Bloque $M_p(6B59) = 6B59 \setminus 8 = {}^{3435}(10;$

675E: Bloque $M_p(675E) = 675E \setminus 8 = {}^{3307}(10$

Aplicando ahora la primera expresión:

6B59: Bloque caché(6B59) = 3435 mod 128 = 107;

675E: Bloque caché(675E) = 3307 mod 128 = 107.

Es decir, a ambos bloques de memoria principal les corresponde el bloque 107 de la caché. Ahora bien, como ambas direcciones pertenecen a bloques de memoria principal diferentes, y como se parte del hecho de que la dirección 6B59 ya se encuentra en la caché, es imposible que también se encuentre simultáneamente la 675E.

2.- [Ver las páginas 53 y ss. y 93-94 del texto base de teoría.] En cualquier jerarquía de memorias, los niveles superiores (más cercanos a la CPU) son más rápidos, más caros y de menor capacidad que los niveles inferiores (Figura 1).

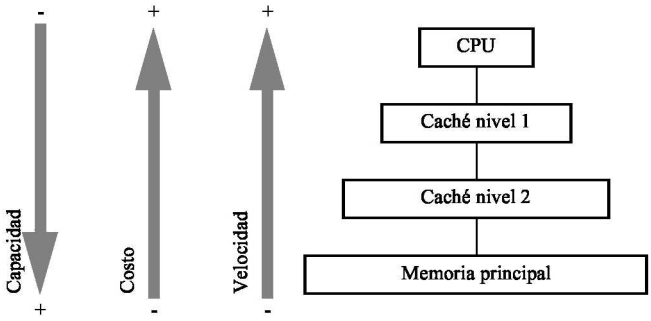


Figura 1: Jerarquía de memorias con dos niveles de caché

I. La caché de nivel 1 es más pequeña que la de nivel 2, por lo que no puede contener una copia de *todos* los bloques de esta última. (No tiene sentido considerar que ambas cachés puedan tener la misma capacidad, pues en una jerarquía de memorias el tamaño disminuye a medida que se está más cerca de los registros de la CPU; hoy en día, una caché de nivel 2 suele ser un orden de magnitud mayor que una caché de nivel 1. [Pág. 94 del texto base de teoría.])

II. [Ver la página 93 del texto base de teoría.] Con la política de post-escritura, las modificaciones que se realicen en la caché de nivel 1 no se reflejarán inmediatamente en la caché de nivel 2. Por tanto, el

contenido de un bloque de caché de nivel 1 *no siempre será igual* al contenido del bloque correspondiente en la caché de nivel 2.

Respuesta: D (I: no, II: no.)

3: [Ver las páginas 68 y ss. del texto base de teoría.] La memoria de este computador tendrá una capacidad de $_2{}^{16}$ = 64 Kpalabras. Como los 48K palabras superiores corresponden a memoria RAM, los 16K palabras inferiores corresponden a la ROM. Con 16 bits de direcciones, las direcciones correspondientes a la ROM son de la forma 00XX XXXX XXXX XXXX (X representa 0 ó 1, indistintamente).

I. Las direcciones de ROM se caracterizan por tener sus dos bits más significativos ($^{\wedge}_{15}$ - $_{A14}$) a 0, mientras que el resto de los bits pueden tomar cualquier valor. Por tanto, la función lógica de selección de la ROM será $^{\wedge}_{15}{}^{\wedge}_{14}$

II. Como se ha visto, las direcciones de ROM son siempre de la forma 00..., mientras que las de RAM pueden ser 01..., 10... ó 11... Por tanto, los dos bits menos significativos (A_1 - A_0) no sirven para distinguir ROM de RAM.

Respuesta: B (I: sí, II: no.)

4: [Ver las páginas 47 y ss. del texto base de teoría.] Al tratarse de una memoria de acceso no aleatorio, el tiempo de acceso a la información dependerá de la posición en la que ésta se encuentre almacenada y de la secuencia de accesos que haya habido anteriormente. Ahora bien, ya que se conoce el tiempo medio que se tarda en colocar la cabeza de lectura - escritura y se conoce la velocidad de transferencia que es capaz de mantener el sistema, es posible calcular el tiempo medio que se tardará en transferir una cantidad determinada de información.

Suponiendo la cabeza ya posicionada, en transferir 10^3 *bytes,* que son 8×10^3 *bits,* se tardará (8×10^3 bits) / (2×10^6 bits/seg) = 4×10^{-3} seg = 4 mseg. Si a esta cantidad se le suman los 2 mseg que se tarda en promedio en posicionar la cabeza, resulta un tiempo medio 6 mseg.

Respuesta: A (6 mseg)

5: [Ver las páginas 77 y ss. del texto base de teoría.] El tiempo medio de acceso viene dado por la expresión:

$$t_A = H \times (\text{Tiempo de acceso en caso de acierto}) + (1\text{-}H) \times (\text{Tiempo de acceso en caso de fallo})$$

donde H es la tasa de acierto (es decir, el tiempo medio de acceso es la media ponderada del tiempo medio de acceso en caso de acierto y del tiempo medio de acceso en caso de fallo).

En caso de acierto, el tiempo medio de acceso es el tiempo de acceso a la memoria caché, 20 nseg. La probabilidad de acierto es del 90 % (0.9).

En caso de fallo, primero se mueve el bloque completo que ha producido el fallo desde la memoria principal a la memoria caché y luego, según el enunciado, se mueve la palabra que ha producido el fallo desde la caché a la CPU. Luego en caso de fallo hay que mover 8 palabras (1 partición o bloque) desde la memoria principal, lo que lleva 8×200 nseg = 1600 nseg; después se mueve una palabra desde la caché, lo que emplea otros 20 nseg. Luego en caso de fallo se tarda 1620 nseg. La probabilidad de un fallo de referencia es del 100 % - 90 % = 10 % (0.1).

El tiempo medio de acceso de este sistema de memoria será por tanto (0.9×20 nseg) + (0.1×1620 nseg) = 18 nseg + 162 nseg = 180 nseg.

Respuesta: B (180 nseg)

6: [Ver las páginas 20 y ss. del texto base de teoría.] Veamos cada una de las afirmaciones por separado:

A) Falsa, pues también transimiten señales de control. [Pág. 22]

B) Verdadera. [Pág. 25]

C) Falsa, pues al tratarse de un medio compartido al que sólo puede acceder un dispositivo en cada ocasión hace falta un método de arbitraje del mismo. [Pág. 27]

Respuesta: B

7.- [Ver las páginas 136 y ss. del texto base de teoría.] En E/S controlada por programa la CPU pide un dato, entra en un bucle de espera y lee el dato cuando el periférico lo tiene disponible.

Según el enunciado de esta pregunta, en un momento dado el computador se encuentra realizando únicamente E/S controlada por programa. Desde que la CPU pide el dato hasta que está disponible pasan 6 mseg. Durante este tiempo la CPU se encuentra ejecutando el bucle de espera. Éste consta de una única instrucción, instrucción que tarda en ejecutarse 200 mseg. Por tanto, el bucle de espera se ejecutará (6 mseg) / (200 mseg) = (6000 mseg) / (200 mseg) = 30 veces.

En leer el dato, ya disponible, y preparar la solicitud de lectura del dato siguiente la CPU emplea otras 10 instrucciones. Luego el proceso completo de pedir y leer un dato dura 40 instrucciones, de las que 30 están dedicadas en exclusiva al bucle de espera. Es decir, el bucle de espera ocupa el 30 / 40 ¥ 100 = 75 % del tiempo de la CPU.

Respuesta: C (75 %)

8.- La manera más sencilla de contestar a esta pregunta consiste en convertir los dos números BCD a su representación en decimal: X = 935 e Y = 151. El cociente de su división entera es 6 con resto 29, resto que expresado como un número *binario* de 12 bits es 000000011101.

Respuesta: C (000000011101)

Cuestiones teórico-prácticas - Junio 2ª semana:

1.- (0.75 puntos) Justificar razonadamente el resultado de la pregunta 1 del test.

2.- (0.5 puntos) Justificar razonadamente el resultado de la pregunta 2 del test.

3.- (0.75 puntos) Justificar razonadamente el resultado de la pregunta 3 del test.

Solución

1.- Esta pregunta ya ha sido contestada a la hora de resolver el test.

2.- Esta pregunta ya ha sido contestada a la hora de resolver el test.

3.- Esta pregunta ya ha sido contestada a la hora de resolver el test.

Problema - Junio 2ª semana:

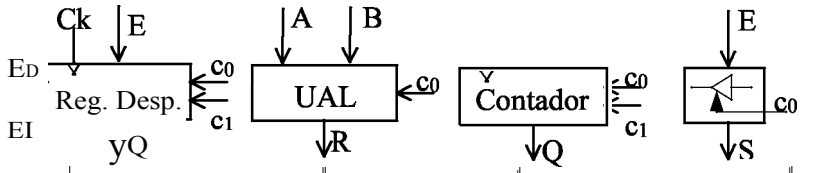
El siguiente algoritmo describe una determinada operación de un sistema digital.

A) (2 puntos) Diseñar la Unidad de Procesamiento que permita realizar este algoritmo utilizando los módulos de la Figura 2: registros de desplazamiento de 8 bits, una UAL con dos entradas de 8 bits cada una, un contador módulo-16 bidireccional y circuitos triestado de conexión unidireccional con control de 8 bits; además de puertas lógicas y los módulos combinacionales (MUX, DMUX, codificadores y decodificadores) que considere necesarios. Debe tener en cuenta que al bus vuelcan datos múltiples

dispositivos y evitar en su diseño posibles conflictos eléctricos entre ellos.

B) (2 puntos) Diseñar la Unidad de Control que ejecute este algoritmo con la Unidad de Procesamiento diseñada en el apartado A) empleando la técnica de los elementos de retardo. **Detalley explique claramente** todos y cada uno de los pasos seguidos hasta obtener la solución.

```
1: Declaración: A[8], B[8], Cont[4]; Bus[8]
2:      A ← Bus;
3:      B <- Bus, Cont = 0;
4:      while Cont ≠ 14
5:          if A es multiplo de 4 then
6:              A ← A - B, Cont = (Cont +2) mod 16;
7:          else
8:              B <- B + A;
9:          endif;
10:     endwhile;
11:     Bus ← B;
12:     Bus ← A;
13:     Parar;
```



c_1c_0	Q(t+1)	c_0	R	c_1c_0	Q(t+1)	c_0	S
00	Q(t)	0	A+B	00	Q(t)	0	-
01	D. Dcha.	1	A-B	01	Q(t)-1 mod 16	1	E
10	D. Izq.			10	Q(t)+1 mod 16		
11	Carga			11	0		

Figura 2: Módulos secuenciales del problema con sus tablas de funcionamiento

---

Solución

a) En primer lugar se deben analizar las operaciones a realizar para ver qué recursos se necesitan y si éstos se encuentran disponibles:

- Sendos registros **A** y **B** de 8 bits que deben poder cargarse desde el bus y volcar su contenido al bus. Así mismo, deben poder cargarse desde la salida de la UAL. Como registros nos ofrece el enunciado registros de desplazamiento de 8 bits con capacidad de carga en paralelo. Éstos son los registros que habrá que utilizar, aunque su capacidad de desplazamiento no se utilizará.
- Para seleccionar el origen de datos de los registros se utilizarán multiplexores.

- Los registros volcarán su contenido al bus a través de sendas **puertas triestado,** para evitar problemas eléctricos.
- Un **contador** módulo-16 (de 4 bits de longitud de palabra por tanto). Aunque en el algoritmo propuesto la cuenta se ha de incrementar en 2 en cada ocasión y el contador de que se dispone sólo puede contar de 1 en 1, esto no supone nungún problema ya que bastará con incrementar 1 dos veces seguidas.

Del análisis del algoritmo se sigue que harán falta dos señales de condición:

- s_0, que indique si el valor del registro A es múltiplo de 4. Para ello veamos cómo son los números múltiplos de 4:

	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
4:	0	0	0	0	0	1	0	0
8:	0	0	0	0	1	0	0	0
12:	0	0	0	0	1	1	0	0
16:	0	0	0	1	0	0	0	0
20:	0	0	0	1	0	1	0	0

Tabla 1: Los múltiplos de 4

Es decir, son todos ellos números acabados en 00. Si se admite el 0 como múltiplo de 4 (este problema se resolverá aquí adoptando este criterio) entonces se tiene que $s_0 = A_1 A_0$. Si se considera que 0 no es múltiplo de 4 entonces alguno de los bits más significativos ha de ser distinto de 0: $s_0 = (A_7 + A_6 + A_5 + A_4 + A_3 + A_2) A_1 A_0$.

Otra manera de ver cómo son los números múltiplos de 4 es la siguiente: Un número binario $a_n a_{n-1} a_{n-2} \ldots a_1 a_0$ tiene el valor decimal $a_n \times 2^n + a_{n-1} \times 2^{n-1} + a_{n-2} \times 2^{n-2} + \ldots + a_1 \times 2^1 + a_0 \times 2^0$. Para dividir este número entre 4 lo multiplicamos por 2^{-2}, con lo que resulta: $a_n \times 2^{n-2} + a_{n-1} \times 2^{n-3} + a_{n-2} \times 2^{n-4} + \ldots + a_2 \times 2^0 + a_1 \times 2^{-1} + a_0 \times 2^{-2}$. Si el número inicial es divisible entre 4, la parte decimal del cociente de la división debe ser 0. En este caso, la parte fraccionaria es $a_1 \times 2^{-1} + a_0 \times 2^{-2}$, pues $2^{-1} = 0.5$ y $2^{-2} = 0.25$. Ya que a_1 y a_0 sólo pueden tomar los valores 0 ó 1 (>= 0), la única forma de que la parte fraccionaria sea 0 es que $a_1 = a_0 = 0$.

- s_1, que indica si el contador vale 14. Con cuatro dígitos binarios, 14 se expresa como 1 1 1 0, luego esta señal de condición se realizará fácilmente con una puerta AND y un inversor.

Un posible diseño para la Unidad de Procesamiento, que cumple con todos estos requisitos, es el que se muestra en la Figura 3.

Ésta no es la única Unidad de Procesamiento capaz de realizar el algoritmo pedido. Por ejemplo, dado que los registros A y B no se van a cargar simultáneamente, es posible hacer uso de un único multiplexor. Así mismo, dado

que la capacidad de desplazamiento de los registros no se va a utilizar, y ya que 11 implica carga en paralelo y 00 implica no-operación, es posible gobernar cada uno de los registros con una única señal de control conectada simultáneamente a sus entradas c_0 y c_1. La Figura 4 muestra cómo se realizarían estas modificaciones. (Este diseño, al hacer uso de menos componentes y menos señales de control, es más económico.) En lo que sigue, el problema se resolverá haciendo uso del diseño de la Figura 3.

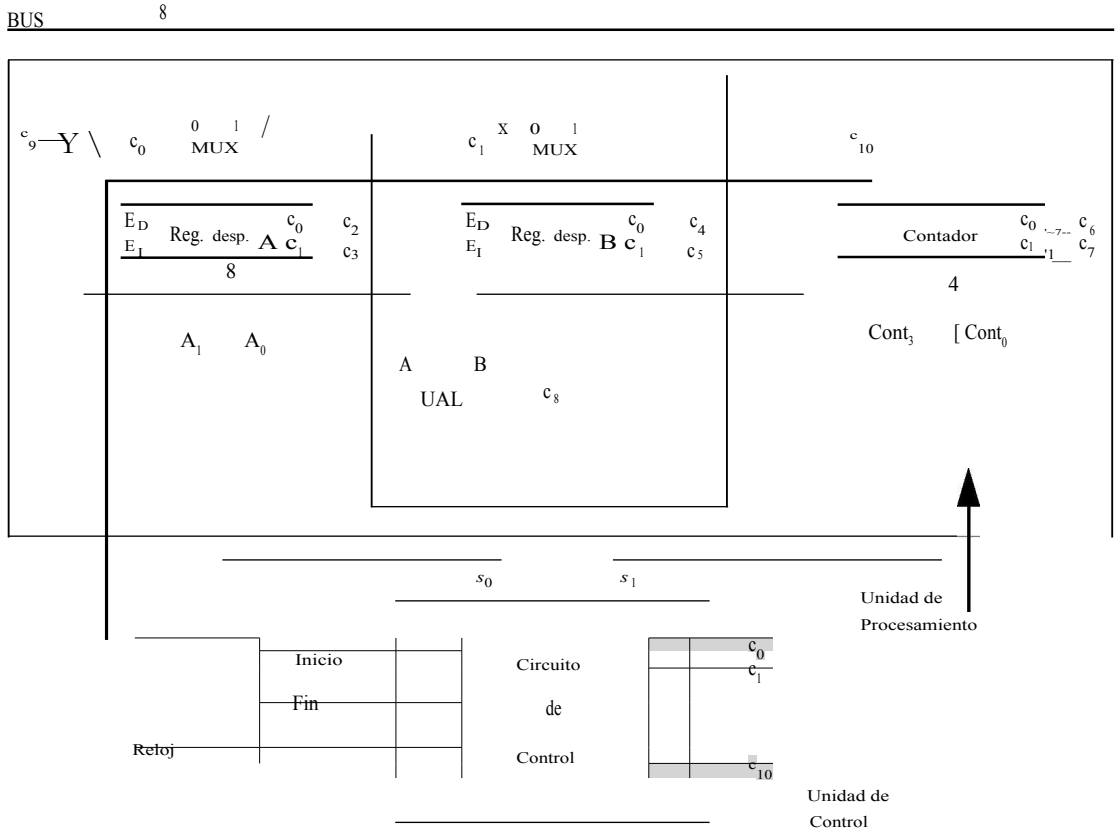


Figura 3: Diseño de la Unidad de Procesamiento

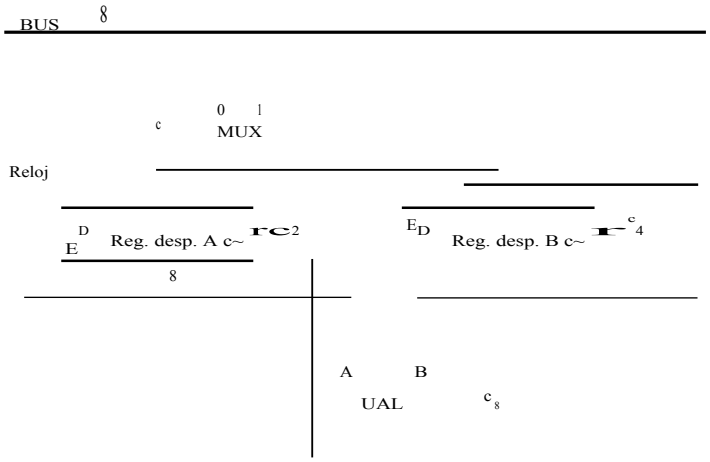


Figura 4: Fragmento de la Unidad de Procesamiento, en el que se muestran algunas posibles modificaciones en el diseño

b) Para describir el funcionamiento de la Unidad de Control solicitada en este apartado, se diseña el diagrama de transición de estados de la Figura 5, donde el significado detallado de cada uno de los estados propuestos viene dado por la Tabla 2. Obsérvese que esta Unidad de Control se ha diseñado como una máquina de Moore. (Dada la sencillez del algoritmo propuesto, se ha obtenido directamente el diagrama de estados a partir de éste, sin necesidad de pasar por el diagrama ASM.)

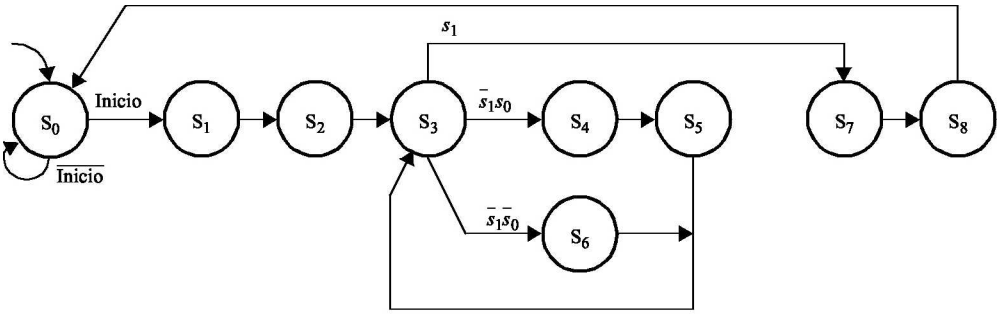


Figura 5: Diagrama de estados de la Unidad de Control

Estado de la Unidad de Control	Microoperaciones efectuadas	Señales de control a activar
S_0	Ninguna	Ninguna
S_1	A <- Bus	(c_0=0), c_2, c_3
S_2	B <- Bus Cont <- 0	(c_1=0), c_4, c_5 c_6, c_7
S_3	Ninguna	Ninguna
S_4	A <- A - B Cont <- (Cont + 1) mod 16	c_0, c_2, c_3, c_8 (c_6=0), c_7
S_5	Cont <- (Cont + 1) mod 16	(c_6=0), c_7
S_6	B <- B + A	c_1, c_4, c_5, (c_8=0)
S_7	Bus <- B	c_{10}
S_8	Bus <- A	c_9

Tabla 2: Acciones tomadas por la Unidad de Control en cada estado

Este diagrama de transición de estados cumple todos los requisitos para ejecutar el algoritmo propuesto en el enunciado del problema utilizando la Unidad de Procesamiento diseñada en el apartado anterior. Se puede comprobar, por ejemplo, que ejecuta el bucle *while* mientras el valor del contador sea distinto de 14, ya que de verificarse esta condición (s_1 = 1) del estado S_3 se salta al estado S_7. En caso contrario, en S_3 se comprueba si A es múltiplo de 4 (s_0) o no (s_0) y se salta al estado correspondiente. Obsérvese que, como se comentó en el apartado anterior, el contador no se puede incrementar en 2 en un único estado, por lo que ha habido que incrementarlo en una unidad dos veces consecutivas, en los estados S_4 y S_5.

La Unidad de Control se realiza utilizando la técnica de los *elementos de retardo,* según pide el enunciado, asignando un biestable tipo D a cada estado, tal como se muestra en la Figura 6. Para evaluar las condiciones en S_3, en lugar de utilizar un demultiplexor de dos entradas de control (s_1 y s_0) y cuatro salidas se ha preferido utilizar, de manera equivalente, dos multiplexores colocados en cascada, cada uno de ellos con una única entrada de control y dos salidas. También se muestra cómo se forman las distintas señales de control a partir de las salidas de los biestables tipo D [ver las páginas 304 a 307 del texto base de teoría]. Obsérvese que los bloques de decisión corresponden a demultiplexores, según se muestra en la Figura 7 [ver las páginas 304 a 307 del texto base de teoría].

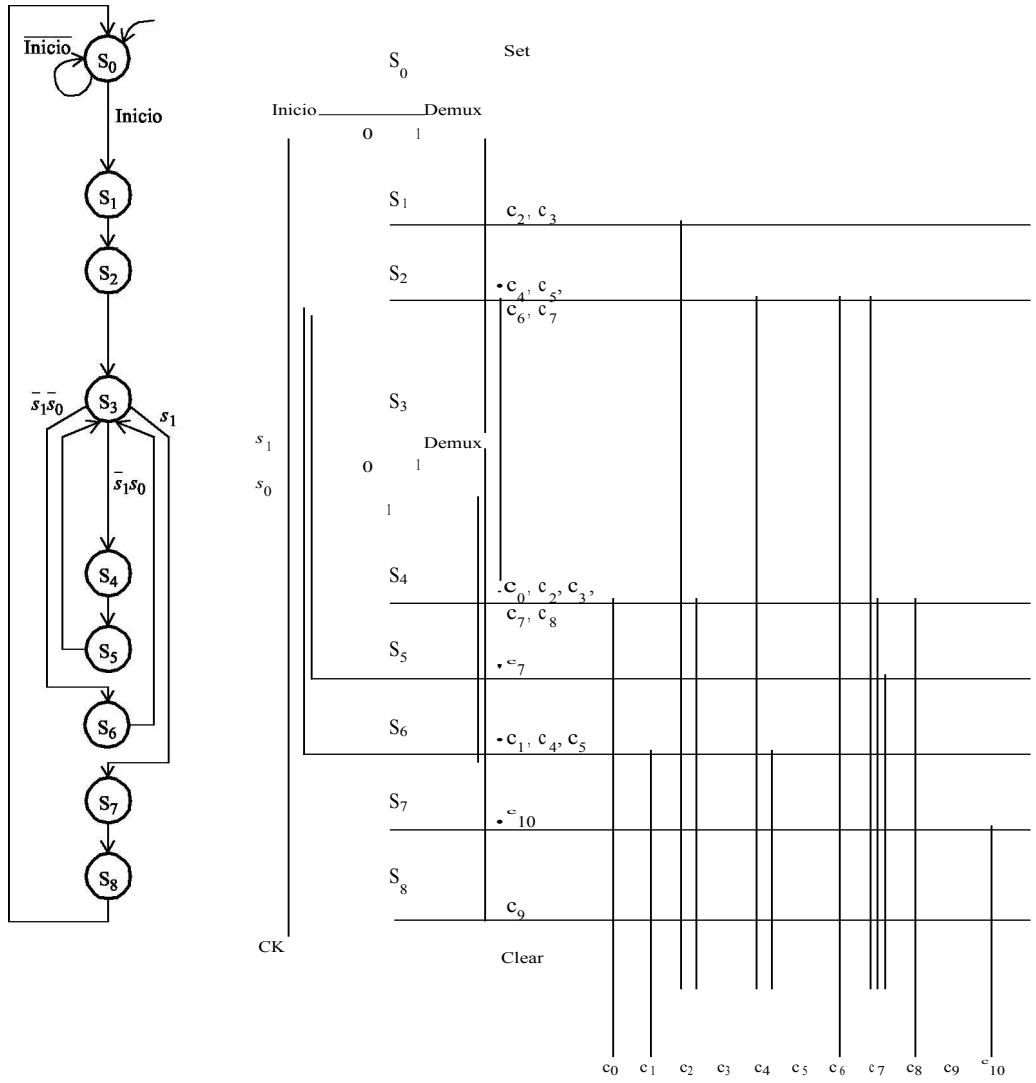


Figura 6: Unidad de Control mediante *elementos de retardo* (derecha); se muestra nuevamente el diagrama de estados (izquierda) para ilustrar el alto grado de paralelismo que existe entre éste y la realización mediante elementos de retardo

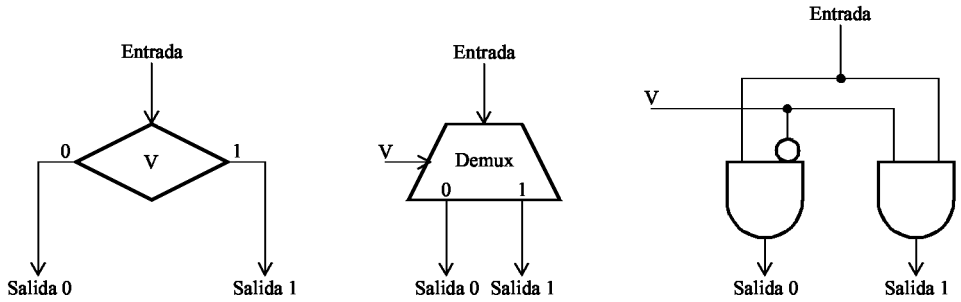


Figura 7: Bloque de decisión (izquierda) visto como un demultiplexor (centro) y su realización equivalente con puertas lógicas (derecha)