

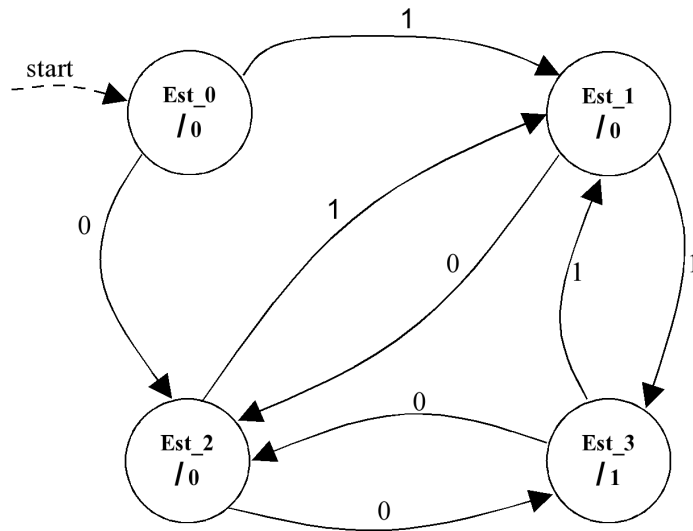
# ESTRUCTURA Y TECNOLOGÍA DE COMPUTADORES

*I.T.I. Gestión, I.T.I. Sistemas e I. Superior, Grupos I y II*

*EXAMEN FINAL, 13 de febrero de 2009*

Apellidos:	Nombre:
D.N.I.:	
Grupo: <input type="checkbox"/> Sistemas I <input type="checkbox"/> Sistemas II <input type="checkbox"/> Gestión I <input type="checkbox"/> Gestión II <input type="checkbox"/> Superior	

1. Dado el siguiente autómata finito determinista representado por un grafo dirigido, se pide:
- (0,2 puntos) Determinar si se trata de una máquina de Mealy o de Moore ¿Por qué?
  - (1 punto) Implementar el sistema secuencial síncrono correspondiente con biestables tipo D disparados por flanco descendente.
  - (0,6 puntos) Determinar la frecuencia máxima a la que puede funcionar el circuito, sabiendo que el retardo de una puerta NOT es de 5 nseg, el de una puerta OR o AND es de 10 nseg y que el tiempo de carga de un biestable es de 20 nseg.
  - (0,2 puntos) ¿Qué función realiza el circuito (para qué sirve)?



2. Dado el siguiente procedimiento escrito en ensamblador MIPS y el siguiente contenido de memoria:

<pre> ; L es 0x10010000 la \$t0, L li \$s0, 0 bucle: beq \$t0, \$0, fin       lw \$t1, 4(\$t0)       add \$s0, \$s0, \$t1       lw \$t0, 0(\$t0)       j bucle fin: </pre>	<table style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">Dirección</th> <th style="text-align: left;">Valor</th> </tr> </thead> <tbody> <tr><td>0x10010000</td><td>0x10010008</td></tr> <tr><td>0x10010004</td><td>0x00000003</td></tr> <tr><td>0x10010008</td><td>0x10010020</td></tr> <tr><td>0x1001000c</td><td>0xffffffff</td></tr> <tr><td>0x10010010</td><td>0x00000000</td></tr> <tr><td>0x10010014</td><td>0x00000004</td></tr> <tr><td>0x10010018</td><td>0x10010010</td></tr> <tr><td>0x1001001c</td><td>0x00000006</td></tr> <tr><td>0x10010020</td><td>0x10010018</td></tr> <tr><td>0x10010024</td><td>0x00000008</td></tr> <tr><td>0x10010028</td><td>0x00000000</td></tr> <tr><td>0x1001002c</td><td>0x00000000</td></tr> </tbody> </table>	Dirección	Valor	0x10010000	0x10010008	0x10010004	0x00000003	0x10010008	0x10010020	0x1001000c	0xffffffff	0x10010010	0x00000000	0x10010014	0x00000004	0x10010018	0x10010010	0x1001001c	0x00000006	0x10010020	0x10010018	0x10010024	0x00000008	0x10010028	0x00000000	0x1001002c	0x00000000
Dirección	Valor																										
0x10010000	0x10010008																										
0x10010004	0x00000003																										
0x10010008	0x10010020																										
0x1001000c	0xffffffff																										
0x10010010	0x00000000																										
0x10010014	0x00000004																										
0x10010018	0x10010010																										
0x1001001c	0x00000006																										
0x10010020	0x10010018																										
0x10010024	0x00000008																										
0x10010028	0x00000000																										
0x1001002c	0x00000000																										

- (0,25 puntos) ¿Cuál será el contenido del registro \$s0 al final de la ejecución del programa?
- (0,5 puntos) Calcule el tiempo de ejecución del programa anterior en un procesador multiciclo similar al descrito en clase con una frecuencia de reloj de 166 Mhz.
- (0,25 puntos) Suponiendo que el programa se ubica en memoria a partir de la dirección 0x04000000, escriba la codificación en código máquina del programa completo. Puede inventar los códigos de operación y de función de las instrucciones. Muestre el resultado en forma de tabla, con una fila por cada instrucción de código máquina (no pseudoinstrucción). Cada fila debe incluir la dirección, la instrucción que se codifica, el formato usado y los campos correspondientes al formato con los valores especificados en binario, hexadecimal o decimal. Los registros \$t0, \$t1 y \$s0 corresponden a \$8, \$9 y \$16, respectivamente.
- (0,5 puntos) Calcule cuánto tardaría en ejecutarse el programa en un procesador monociclo (similar al descrito en clase) si suponemos que el tiempo de acceso a registro del procesador monociclo es igual al tiempo de ciclo del multiciclo del apartado a, y el tiempo de acceso a memoria es el cuádruple que el tiempo de acceso a los registros, que es igual que la latencia de la ALU.
- (0,5 punto) Convierta el programa anterior en un procedimiento que reciba la dirección de inicio como un argumento (en lugar de ejecutar la primera instrucción para cargarla) y devuelva el resultado al procedimiento que lo llama (en lugar de almacenarlo en \$s0). Respete todas las convenciones de MIPS sobre el uso de los registros.

3. (2 puntos) Se desea diseñar una memoria virtual para un procesador que cumpla los siguientes requisitos:
- Direcciones físicas de 24 bits y memoria direccionable por bytes.
  - La consulta de la tabla de páginas debe realizarse en un único acceso a memoria. Además, la tabla de páginas no puede ocupar más del 6,25% (la dieciseisava parte) de la capacidad máxima de la memoria física. Cada entrada de la tabla de páginas tiene un tamaño de 2 bytes.
  - Para los fallos de página se empleará una política de reemplazo pseudo-LRU (un sólo bit de uso) y una estrategia de post-escritura.

Se pide:

- Según las especificaciones anteriores, diseñar el formato óptimo de las direcciones virtuales y físicas y de la tabla de páginas si se desea implantar una memoria virtual con la mayor capacidad posible. Especificar detalladamente el formato de la dirección física y de la dirección virtual, dibujar un esquema de la tabla de páginas detallando los bits de control necesarios (justificando su inclusión).
- Se desea implantar también un TLB 2-asociativo de dos conjuntos (4 bloques en total) con estrategia de reemplazo LRU y una política de post-escritura. Dibujar un esquema detallado del TLB y calcular el tamaño exacto incluyendo los bits de control necesarios (justificando su inclusión).
- Mostrar de forma exacta cómo evoluciona el contenido del TLB y de la tabla de páginas después de cada una de las siguientes referencias virtuales generadas por el procesador expresadas en **hexadecimal**: lectura 810, lectura 1010, escritura 800 y escritura 2000. Obtener también la dirección física resultante. (Nota: para los fallos de página, suponga que las primeras direcciones bajas de memoria están libres.)

# SOLUCIÓN 1

a) Determinar si se trata de una máquina de Mealy o de Moore ¿Por qué?

Se trata de una máquina Moore pues las salidas están asociadas a los estados y no a las transiciones. O lo que es lo mismo las salidas dependen sólo del estado actual dado por la memoria de estado.

b) Implementar el sistema secuencial síncrono correspondiente con biestables tipo D disparados por flanco descendente.

Su tabla de estados es:

Estado actual	E=0	E=1	SALIDA
EST_0	EST_2	EST_1	0
EST_1	EST_2	EST_3	0
EST_2	EST_3	EST_1	0
EST_3	EST_2	EST_1	1

Realizando la asignación de estados, deducimos la tabla de transiciones (existen otras combinaciones de valores válidas, por lo tanto otras resoluciones para el ejercicio):

Estado actual $Q_1 Q_0$	E=0		E=1		SALIDA
	$Q_1^* Q_0^*$	$Q_1^* Q_0^*$	$Q_1^* Q_0^*$	$Q_1^* Q_0^*$	
EST_0 $\Rightarrow$ 0 0	1 0	0 1	0		
EST_1 $\Rightarrow$ 0 1	1 0	1 1	0		
EST_2 $\Rightarrow$ 1 0	1 1	0 1	0		
EST_3 $\Rightarrow$ 1 1	1 0	0 1	1		

$D_1$

$Q_1 Q_0$	00	01	11	10
0	1 <sub>0</sub>	1 <sub>1</sub>	1 <sub>3</sub>	1 <sub>2</sub>
1	0 <sub>4</sub>	1 <sub>5</sub>	0 <sub>7</sub>	0 <sub>6</sub>

$D_0$

$Q_1 Q_0$	00	01	11	10
0	0 <sub>0</sub>	0 <sub>1</sub>	0 <sub>3</sub>	1 <sub>2</sub>
1	1 <sub>4</sub>	1 <sub>5</sub>	1 <sub>7</sub>	1 <sub>6</sub>

$$D_1 = E' + Q_1' \cdot Q_0$$

$$D_0 = E + Q_1 \cdot Q_0'$$

$Q_1$	$Q_0$	S
0	0	0 <sub>0</sub>
0	1	0 <sub>1</sub>
1	0	0 <sub>2</sub>
1	1	1 <sub>3</sub>

$$S = Q_0 \cdot Q_1$$

El circuito, como suma de productos, quedaría:



ori \$s0, \$0, 0	1	4	4
bucle: beq \$t0, \$0, fin	6	3	18
lw \$t1, 4(\$t0)	5	5	25
add \$s0, \$s0, \$t1	5	4	20
lw \$t0, 0(\$t0)	5	5	25
j bucle	5	3	15
	28		111

Del dato de la frecuencia, el tiempo de ciclo es  $1 / 166 \text{ Mhz} = 6.0241 \text{ ns}$ .  
Por tanto, el tiempo de ejecución es:  $6.0241 \text{ ns} \times 111 \text{ ciclos} = 668.675 \text{ ns}$ .

**c)**

De nuevo, hay que tener en cuenta que las instrucciones a codificar incluyen las siguientes pseudoinstrucciones:

la \$t0, 0x10010008 → lui \$t0, 0x1001

li s0, 0 → ori \$s0, \$0, 0

La codificación resultante es la siguiente:

Dirección	Instrucción	Formato	Código máquina			
0x04000000	lui \$t0, 0x1001	I	lui	0	8	0x1001
0x04000004	ori \$s0, \$0, 0	I	ori	0	16	0
0x04000008	beq \$t0, \$0, fin	I	beq	8	0	4
0x0400000c	lw \$t1, 4(\$t0)	I	lw	8	9	4
0x04000010	add \$s0, \$s0, \$t1	R	0	16	9	16 0 add
0x04000014	lw \$t0, 0(\$t0)	I	lw	8	8	0
0x04000018	j bucle	J	j	0x1000001		

**d)**

De la solución del apartado a):

$$T_{\text{ciclo}}(\text{multi}) = 6.0241 \text{ ns}$$

$$N_{\text{inst}} = 28 \text{ instrucciones}$$

De los datos del problema:

$$T_{\text{BR}} = T_{\text{ciclo}}(\text{multi}) = 6.0241 \text{ ns}$$

$$T_{\text{Mem}} = 4 \times T_{\text{BR}}$$

$$T_{\text{ALU}} = T_{\text{BR}}$$

Además, sabemos que en general:

$$T_{\text{ciclo}}(\text{mono}) = T_{\text{Mem}} + T_{\text{BR}} + T_{\text{ALU}} + T_{\text{Mem}} + T_{\text{BR}} = 2 \times T_{\text{Mem}} + 2 \times T_{\text{BR}} + T_{\text{ALU}}$$

Por tanto, en este caso:

$$T_{\text{ciclo}}(\text{mono}) = 2 \times 4 \times T_{\text{BR}} + 2 \times T_{\text{BR}} + T_{\text{BR}} = 11 \times T_{\text{BR}} = 66.2651 \text{ ns}$$

Por tanto:

$$T_{\text{mono}} = N_{\text{inst}} \times T_{\text{ciclo}}(\text{mono}) = 28 \times 66.2651 \text{ ns} = 1.855 \mu\text{s}$$

**e)**

Hay que tener en cuenta que el procedimiento usa el registro \$s0, por lo que será necesario almacenarlo en la pila. Alternativamente, se puede cambiar el uso del registro \$s0 por el de algún registro temporal.

Debemos también mover la dirección que nos pasan en el registro  $\$a0$  al registro  $\$t0$  (o bien modificar las referencias al registro  $\$t0$  para que usen el registro  $\$a0$ ).

Al final del procedimiento, habrá que copiar el valor de  $\$s0$  a  $\$v0$  para devolvérselo a la rutina que haya llamado. Alternativamente, se puede usar el registro  $\$v0$  directamente para acumular el resultado, ahorrando de esta forma también la necesidad de usar la pila.

El procedimiento quedaría de la siguiente forma:

```

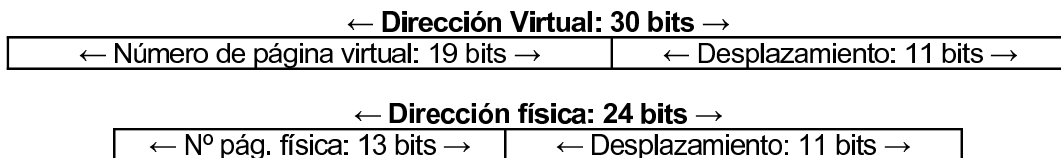
sumar_lista:
    addi $sp, $sp, -4 ; hace sitio en la pila
    sw   $s0, 0($sp) ; guarda s0 en la pila
    move $t0, $a0    ; pone la dirección de inicio en t0
    li   $s0, 0
bucle:
    beq  $t0, $0, fin
    lw   $t1, 4($t0)
    add  $s0, $s0, $t1
    lw   $t0, 0($t0)
    j    bucle
fin:
    move $v0, $s0    ; pone el resultado en v0
    lw   $s0, 0($sp) ; recupera el antiguo valor de s0
    addi $sp, $sp, 4 ; devuelve el espacio de la pila
    jr   $31
    
```

### SOLUCIÓN 3

a)

Los 24 bits de las direcciones físicas de memoria indican que disponemos de una memoria física máxima de 16MB, de los cuales reservamos 1/16 para alojar la tabla de páginas cuyo tamaño será de 1MB. Con ese tamaño tendremos 512K-entradas para la tabla de páginas con dos bytes por entrada. En esos dos bytes se debe codificar el bit de uso (U), el bit de modificación (M) y el bit de validez (V), por lo que quedan 13 bits para el número de página física. El campo desplazamiento serían los 11 bits restantes hasta conseguir los 24 bits totales de la dirección física de memoria. Luego el tamaño de página será de 2KB (11 bits en la dirección física).

Como tenemos 512K-entradas en la tabla de páginas, necesitamos 19 bits para indicar el número de entrada, es decir, el número de página virtual. Luego el tamaño de la memoria virtual sería de 1GB ( $2^{19+11}$ ). El formato de la dirección física y de la dirección virtual sería:



**Tabla de Páginas**

	←1→	←1→	←1→	← 13 →
dirs	<b>V</b>	<b>M</b>	<b>U</b>	<b>nº pag. física</b>
0				
1				
..				
..				
$2^{19}-2$				
$2^{19}-1$				

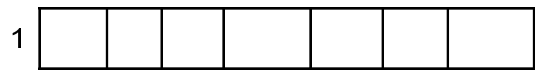
El bit V (validez) siempre se debe incluir para saber si el contenido de una entrada es válido o no. El bit M (modificación) se debe incluir porque para la tabla de páginas se utiliza una estrategia de post-escritura. Finalmente, el bit U (uso) se debe incluir para implementar un reemplazo pseudo-LRU.

b)

#### TLB

	<b>V</b>	<b>U</b>	<b>M</b>	<b>ETI</b>	<b>M'</b>	<b>U'</b>	<b>NPF</b>
0	1	1	1	18	1	1	13

	<b>V</b>	<b>U</b>	<b>M</b>	<b>ETI</b>	<b>M'</b>	<b>U'</b>	<b>NPF</b>
0	1	1	1	18	1	1	13



Tenemos 2 conjuntos y 2 vías, lo que nos permite almacenar en el TLB hasta 4 entradas distintas de la tabla de páginas. Por lo tanto, de los 19 bits que conforman el NPV, necesitamos 1 bit para el índice y nos quedan 18 bits para la etiqueta.

**Tamaño =  $2 * 2 * (3+18+2+13)$  bits = 144 bits**

El bit V (validez) siempre se debe incluir para saber si el contenido de una entrada es válido o no. El bit M (modificación) se debe incluir porque para el TLB se utiliza una estrategia de post-escritura. Finalmente, el bit U (uso) se debe incluir para implementar un reemplazo LRU. Generalmente, un único bit de uso no es suficiente para implementar un reemplazo LRU puro. Sin embargo, en este caso es más que suficiente porque cada conjunto tiene sólo dos entradas.

c)

Para resolver esta parte, debemos tener en cuenta que las direcciones virtuales son de 30 bits, de los que 19 corresponden al número de página virtual (NPV) y 11 al desplazamiento. En el caso del TLB, de los 19 bits que corresponden al NPV, el bit menos significativo es el índice que selecciona el conjunto del TLB y los 18 bits restantes son la etiqueta.

- Lectura  $810_{16} = 00\ 0000\ 0000\ 0000\ 0000\ 1000\ 0001\ 0000_2$ :  
 NPV =  $00\ 0000\ 0000\ 0000\ 0000\ 1$  → Índice = 1, etiqueta = 0  
 Desplazamiento =  $000\ 0001\ 0000$

Fallo de TLB y fallo de página. El S.O. lee de disco la página virtual 1 a la página física 0 (la memoria está libre). Se actualiza la tabla de páginas:

**Tabla de Páginas**

dirs	V	M	U	nº Pag.
0				
1	1	0	0	0
2				
..				
$2^{19}-2$				
$2^{19}-1$				

Se actualiza el TLB:

	V	U	M	ETI	M'	U'	NPF
0							
1	1	1	1	0	0	1	0

	V	U	M	ETI	M'	U'	NPF
0							
1							

La dirección física resultante es:  $0000\ 0000\ 0000\ 0000\ 0001\ 0000_2 = 10_{16}$

- Lectura  $1010_{16} = 00\ 0000\ 0000\ 0000\ 0001\ 0000\ 0001\ 0000_2$ :  
 NPV =  $00\ 0000\ 0000\ 0000\ 0001\ 0$  → Índice = 0, etiqueta = 1  
 Desplazamiento =  $000\ 0001\ 0000$

Fallo de TLB y fallo de página. El S.O. lee de disco la página virtual 2 a la página física 1 (la siguiente página libre en memoria). Se actualiza la tabla de páginas:

**Tabla de Páginas**

dirs	V	M	U	nº Pag.
0				
1	1	0	0	0
2	1	0	0	1
..				
$2^{19}-2$				
$2^{19}-1$				

Se actualiza el TLB:

V	U	M	ETI	M'	U'	NPF

V	U	M	ETI	M'	U'	NPF

0	1	1	1	1	0	1	1
1	1	1	1	0	0	1	0

0							
1							

La dirección física resultante es: **0000 0000 0000 1000 0001 0000**<sub>2</sub> = 810<sub>16</sub>

- **Escritura 800**<sub>16</sub> = **00 0000 0000 0000 0000 1000 0000 0000**<sub>2</sub>:  
 NPV= = **00 0000 0000 0000 0000 1** → Índice = 1, etiqueta = 0  
 Desplazamiento = 000 0000 0000

La información para la traducción se encuentra en el TLB ya que en el segundo conjunto del TLB hay una entrada cuya etiqueta coincide con la obtenida del NPV. Si tenemos un acierto de TLB, obligatoriamente también hay un acierto en la tabla de páginas. Lo único que hay que hacer es actualizar el TLB para indicar que la página se ha modificado, pero no hay que actualizar la tabla de páginas pues el TLB tiene política de post-escritura:

	V	U	M	ETI	M'	U'	NPF
0	1	1	1	1	0	1	1
1	1	1	1	0	1	1	0

	V	U	M	ETI	M'	U'	NPF
0							
1							

La dirección física resultante es: **0000 0000 0000 0000 0000 0000**<sub>2</sub> = 0<sub>16</sub>

- **Escritura 2000**<sub>16</sub> = **00 0000 0000 0000 0010 0000 0000 0000**<sub>2</sub>:  
 NPV= = **00 0000 0000 0000 0010 0** → Índice = 0, etiqueta = 2  
 Desplazamiento = 000 0000 0000

Se produce un fallo de TLB, pues la entrada en el primer conjunto tiene una etiqueta distinta de 2. También se produce un fallo de página al no encontrarse la información en la tabla de páginas. El S.O. lee de disco la página virtual 4 a la página física 2 (la siguiente página libre en memoria). Se actualiza la tabla de páginas:

Tabla de Páginas			
Dir	V	M	Nº Pag.
0			
1	1	0	0
2	1	0	1
3			
4	1	0	2
..			
2 <sup>19</sup> -2			
2 <sup>19</sup> -1			

Se actualiza el TLB:

	V	U	M	ETI	M'	U'	NPF
0	1	0	1	1	0	1	1
1	1	1	1	0	1	1	0

	V	U	M	ETI	M'	U'	NPF
0	1	1	1	2	1	1	2
1							

Al ser el TLB asociativo de 2 vías y al haber en el primer conjunto todavía un bloque libre, no es necesario expulsar nada del TLB. Lo que sí es necesario es poner a 0 el bit de uso del primer bloque del conjunto 0. Esto nos permite saber qué bloque se utilizó hace más tiempo para implementar así una política de reemplazo LRU.

La dirección física resultante es: **0000 0000 0001 0000 0000 0000**<sub>2</sub> = 1000<sub>16</sub>