

# Arquitectura e Ingeniería de Computadores

Prueba del 27/Noviembre/2006. E.P.S, U.A.M.

## SOLUCIONES

**P1** Completar los huecos punteados que se indican en el siguiente código VHDL de tal forma que la salida flag se ponga a '1' cuando se detecta la secuencia "0110". En el resto de casos la salida flag debe estar a 0. El funcionamiento del circuito es el siguiente:

En cada flanco de subida de reloj se envía un dato por la entrada DATA. El circuito tiene una señal RESET asíncrona activa a nivel alto, que sirve para indicar el reset del circuito.

Considerar que en el circuito que modela el proceso no se produce solapamiento.

```
entity DETECTOR_SECUENCIA is
port (
  DATA : in std_logic;      -- Datos de entrada
  RESET : in std_logic;
  CLK   : in std_logic;      -- Reloj
  FLAG  : out std_logic);    -- Salida
end DETECTOR_SECUENCIA;

architecture PRACTICA of DETECTOR_SECUENCIA is

  type estado_t is (IDLE, PRIMER0, PRIMER1, SEGDO1, FLAG_OK);

  signal estado : estado_t := IDLE;

begin -- PRACTICA

  process(clk,reset)
  begin
    if reset='1' then
      estado<=IDLE;
      FLAG<='0';

    elsif rising_edge(clk) then

      -- Segun el estado del automata
      case estado is

        when IDLE=>
          flag<='0';
          if DATA='0' then
            estado<=PRIMER0;
          else
            estado<=IDLE;
          end if;

        when PRIMER0 =>
          flag<='0';
          if DATA='1' then
            estado<=PRIMER1;
          else
            estado<=PRIMER0;
          end if;

        when PRIMER1 =>
          flag<='0';
          if DATA='1' then
            estado<=SEGDO1;
          else
            estado<=PRIMER0;
          end if;

      end case;

    end if;

  end process;

end PRACTICA;
```

## Arquitectura e Ingeniería de Computadores

Prueba del 27/Noviembre/2006. E.P.S, U.A.M.

```
end if;

when SEGDO1 =>
  if DATA='1' then
    estado<=IDLE;
    flag<='0';

  else
    flag<='1';
    estado<=FLAG_OK;      --También sería válida la sentencia estado <= IDLE;
  end if;

when FLAG_OK =>
  flag<='0';
  if DATA='0' then
    estado<=PRIMERO;
  else
    estado<=IDLE;
  end if;

  when others => null;
end case;
end if;
end process;

end PRACTICA;
```

## Arquitectura e Ingeniería de Computadores

Prueba del 27/Noviembre/2006. E.P.S, U.A.M.

**P2.-** Se pretende mejorar un sistema ordenador con memorias de instrucciones y de datos separadas. Las instrucciones de este sistema se pueden agrupar en cuatro tipos: saltos, operaciones entre registros como enteros, operaciones entre registros en coma flotante y operaciones con memoria (load/store). Sólo estas últimas acceden a la memoria de datos, y sólo acceden a un dato cada vez. La tabla representa el porcentaje del tiempo total dedicado por la CPU a cada tipo de instrucción en el sistema original. En la tabla, también se indica el número de ciclos de reloj que consume cada instrucción en la ruta de datos, es decir, excluyendo los accesos a memoria.

	SALTOS	REG. ENT.	REG. C.F.	LD/ST
%Tiempo total de CPU	10%	20%	30%	40%
Ciclos usados en la Ruta de Datos	2	3	15	3

El sistema original incluye una caché de instrucciones con tasa de aciertos del 98%. El tiempo de acceso a caché es de 1 ciclo y el de acceso a memoria es de 5 ciclos. Los bloques son de una palabra.

El sistema original no incluye caché de datos y el tiempo de acceso a la memoria de datos es de 5 ciclos.

Se analizan dos posibles mejoras del sistema.

**1)** Mejorar la unidad de coma flotante. Esta mejora supondría reducir el número de ciclos para la ruta de datos de dichas instrucciones de 15 a 5 ciclos.

**2)** Incluir una caché de datos. La caché sería de post-escritura con una tasa de aciertos del 95% y un porcentaje de bloques modificados del 40%. Los bloques serían de una palabra y el tiempo de acceso a caché de 1 ciclo (el tiempo de acceso a la memoria de datos no varía).

Se pide:

**a)** Si sólo se puede realizar una de las dos mejoras, ¿cuál elegiría? ¿Qué aceleración se conseguiría en dicho caso?

**b)** Si se incluyen ambas mejoras, ¿qué aceleración global se consigue?

### SOLUCIÓN:

El número de ciclos dedicado a acceder a la memoria de instrucciones es, en promedio:

$$N_{\text{caché}} + (1-H) \cdot N_{\text{mem}} = 1 + 0,02 \cdot 5 = 1,1$$

#### Mejora 1

Los CPI de las instrucciones entre registros en coma flotante pasan de  $15 + 1,1$  a  $5 + 1,1$ , es decir, la aceleración de la mejora es:  $A_{m1} = 16,1/6,1 = 2,6393$

La fracción mejora es el 30% ( $F_{m1} = 0,3$ ). Por tanto, la aceleración global para esta mejora es:

$$A_{g1} = 1/(1 - F_{m1} + F_{m1}/A_{m1}) = 1,2290, \text{ una aceleración del } 22,9\%$$

#### Mejora 2

El número de ciclos dedicados a acceder a la memoria de datos tras la mejora será:

$$N_{\text{caché}} + (1-H) \cdot N_{\text{mem}} + (1-H) \cdot w_M \cdot N_{\text{mem}} = 1 + 0,05 \cdot 5 + 0,05 \cdot 0,4 \cdot 5 = 1,35$$

Por tanto, los CPI de las instrucciones LD/ST (las únicas afectadas por la mejora) pasan de  $3 + 1,1 + 5$  a  $3 + 1,1 + 1,35$ , lo que supone una aceleración de  $A_{m2} = 9,1/5,45 = 1,6697$

La aceleración global, teniendo en cuenta que la fracción mejorada es  $F_{m2} = 0,4$ , es de:

$$A_{g2} = 1/(1 - F_{m2} + F_{m2}/A_{m2}) = 1,1911, \text{ una aceleración del } 19,11\%$$

**a)** Se elegiría la mejora 1, consiguiendo una mejora del 22,9%

**b)** Si se aplica la mejora 2 al sistema ya mejorado 1, la aceleración parcial sigue siendo  $A_{m2} = 1,6697$ . Sin embargo, la fracción mejorada es la siguiente una vez que se realiza la mejora 2 sobre el sistema ya mejorado por 1:

$$F_{m2} = 0,4/(0,1 + 0,2 + 0,3/2,6393 + 0,4) = 0,4916$$

Por tanto, realizar la mejora 2 sobre la mejora 1 consigue una aceleración de:

$$A_{g2} = 1/(1 - F_{m2} + F_{m2}/A_{m2}) = 1,2456, \text{ una aceleración del } 19,11\%$$

La aceleración total tras las dos mejoras es:

$$A_g = A_{g1} \cdot A_{g2} = 1,2290 \cdot 1,2456 = 1,5309, \text{ una aceleración global del } 53,09\%$$

Otra forma de llegar al mismo resultado es aplicar ambas mejoras, que son independientes, en la misma fórmula. Para ello se utilizan los datos de cada mejora independientemente de la otra (ver arriba):

$$A_g = 1/(1 - F_{m1} - F_{m2} + F_{m1}/A_{m1} + F_{m2}/A_{m2}) = 1/(1 - 0,3 - 0,4 + 0,3/2,6393 + 0,4/1,6697) = 1,5309$$

## Arquitectura e Ingeniería de Computadores

Prueba del 27/Noviembre/2006. E.P.S, U.A.M.

**P3.-** La caché de un cierto sistema es unificada, asociativa de dos vías (CA2V), 16 entradas por vía y cuatro palabras por bloque. En un momento determinado, justo tras finalizar la instrucción Clear R2, la caché presenta el estado que se muestra en el esquema adjunto, donde se marcan con XXXX el dato que cada bloque contiene. Además asociado a cada bloque, se indican el estado del contador (1bit) utilizado para el algoritmo LRU implementado siguiendo el sistema estudiado en clase. A continuación se ejecuta el bucle 5 veces y se termina con la ejecución de la instrucción Halt. **Nota:** El sistema sólo maneja instrucciones y datos de 32 bits.

PC				
0F00A4		Clear	R2	; 0 → R2
0F00A8	Bucle:	Load	R1, R2, #7000	; Mem [(R2) + 007000] → R1
0F00AC		Add	R3, R1, R5	; R1 + R5 → R3
0F00B0		Add	R2, R2, #4	; R2 + 4 → R2
0F00B4		Bne	R2, #20, Bucle	; Si R2 ≠ 20 <sub>10</sub> , salta a bucle
0F00B8		Jmp	Fin	; Salta a Fin
-----				
-----				
F00500	Fin:	Halt		; Finaliza la ejecución

VIA 1			VIA 2		
DIRECTORIO	BLOQUES	LRU	DIRECTORIO	BLOQUES	LRU
0F00	XXXX	1	F000	XXXX	0
00F0	XXXX	0	0000	XXXX	1
000F	XXXX	0	000F	XXXX	1
0000	XXXX	1	00F0	XXXX	0
F000	XXXX	1	0F00	XXXX	0
0F00	XXXX	0	F000	XXXX	1
00F0	XXXX	0	0000	XXXX	1
000F	XXXX	1	000F	XXXX	0
0000	XXXX	1	00F0	XXXX	0
F000	XXXX	0	0F00	XXXX	1
0F00	XXXX	0	F000	XXXX	1
00F0	XXXX	1	0000	XXXX	0
000F	XXXX	1	000F	XXXX	0
0000	XXXX	0	00F0	XXXX	1
F000	XXXX	0	0F00	XXXX	1
0F00	XXXX	1	F000	XXXX	0

Se pide, con los datos de los que se dispone y justificando cada una de las respuestas:

**a) Los campos en los que se divide la dirección para el acceso a la caché.**

$2^2$  palabras/bloque \*  $2^2$  bytes/palabra =  $2^4$  bytes por bloque. Se necesitan 4 bits para su selección.

Si la caché tiene  $2^2$  entradas por vía Se necesitan 4 bits para seleccionar una de las entradas.

Si como vemos en el código, las direcciones son de  $2^{24}$  bits, nos quedan 16 bits para la etiqueta.

ETIQUETA	INDICE	BYTE EN BLOQUE
16	4	4

**b) El tamaño de la caché, señalando los bytes de datos, los bits asociativos y los comparadores de los que dispone.**

a1) La caché guarda un total de  $2 \text{ vías} * 2^4 \text{ bloques/vía} * 2^4 \text{ bytes/bloque} = 2^9$  bytes de datos.

a2) La caché dispone de un total de  $2 \text{ vías} * 2^4 \text{ entradas/vía} * 2^4 \text{ bits/entrada} = 2^9$  bits asociativos.

a3) La caché necesita un total de 2 comparadores cada uno de 16 bits.

Se necesita un comparador por cada vía

**c) La tasa de aciertos para el código ejecutado.**

A 4 direcciones de código, 0F00A8:0F00B4 accede 5 veces sólo falla la primera vez 0F00B0.

A 2 direcciones de código 0F00B8 y F00500, accede sólo 1 vez, sólo falla en la F00500.

## Arquitectura e Ingeniería de Computadores

Prueba del 27/Noviembre/2006. E.P.S, U.A.M.

A las 5 direcciones de datos 007000:007010, accede sólo 1 vez y falla en la 007000 y 007010.

Hay por tanto 27 accesos y 4 fallos. Es decir una tasa de aciertos del  $(23/27 = 0,85)$  85%.

- d) El estado de la caché tras la ejecución, incluyendo los contadores del algoritmo. Utilizando el esquema facilitado, señalar sólo aquellas entradas que hayan variado durante la ejecución, indicando el valor del índice de la entrada correspondiente y los valores para la etiqueta y contador LRU finales.

Dirección de acceso:

ETQ	IND	B/B	A/F	Explicación:
0F00	A	8	A	Acierto, índice A, vía 1.
0070	0	0	F	Fallo. Carga un bloque índice 0, vía 1 (LRU). Cambian los contadores de ambas vías.
0F00	A	C	A	Acierto, índice A, vía 1.
0F00	B	0	F	Fallo. Carga un bloque índice B, vía 2 (LRU). Cambian los contadores de ambas vías.
0F00	B	4	A	Acierto, índice B, vía 2.
0F00A8:0F00B4			A	16 aciertos en el acceso a código de las 4 iteraciones.
007004:00700C			A	3 aciertos en el acceso a datos, índice 0, vía 1. 3 iteraciones
0070	1	0	F	Fallo. Carga un bloque índice 1, vía 1 (LRU). Última iteración
0F00	B	8	A	Acierto, índice B, vía 2.
F005	0	0	F	Fallo. Carga un bloque índice 0, vía 2 (LRU). Cambian los contadores de ambas vías.

INDICE	VIA 1			VIA 2		
	DIRECTORIO	BLOQUES	LRU	DIRECTORIO	BLOQUES	LRU
0	0070	XXXX	1	F005	XXXX	0
1	00F0	XXXX	1	0070	XXXX	0
B	0F00	XXXX	0	0000	XXXX	1

## Arquitectura e Ingeniería de Computadores

Prueba del 27/Noviembre/2006. E.P.S, U.A.M.

**P4.-** Se dispone de un ordenador con un procesador cuyas direcciones virtuales son de 48 bits y que dispone de una memoria real de 4 Gbyte. Se opta por un sistema de memoria paginado, con un tamaño de página de 64 Kbytes y longitud del descriptor de página de 4 bytes en el que se incluye, el marco de página, un bit de presencia y varios bits de control. Como la tabla de páginas es muy grande, ésta a su vez está paginada en tres niveles y sólo se guarda una parte en memoria real (los dos últimos niveles). La unidad de manejo de memoria (MMU) del sistema dispone de una unidad TLB 4-asociativo de 128 posiciones. Además del TLB, dispone de una memoria de sustitución directa, que contiene todos los descriptores del primer nivel de las páginas en las que se ha dividido la tabla de páginas. Se pide, justificando la respuesta, señalar:

- a) Campos en que se divide la dirección virtual para el acceso al TLB y para el acceso a las tablas, la dirección real, los campos del TLB y los descriptores de página
- b) El tamaño del TLB y de la memoria de sustitución directa presente en la MMU
- c) El tamaño total de las tablas de páginas (solo nivel 2 y nivel 3). Si todas las páginas estuviesen en memoria que porcentaje representan respecto del total de memoria.
- d) Cuantos accesos a memoria principal se deben realizar para obtener un dato, si la dirección no está en el TLB

a) DV: 48 bits; DR: 20 bits; Página: 64 KBytes =  $2^{16}$  Bytes => 16 bits de offset

En una tabla de páginas, del tamaño de una página, se pueden guardar  $2^{16} / 2^2 = 2^{14}$  descriptores

Si hay 2 niveles de tabla en memoria queda para el primer nivel:

$$48 \text{ (DV)} - 16 \text{ (OFFSET)} - 14 \text{ (N3)} - 14 \text{ (N2)} = 4 \text{ bits para el nivel 1 (N1).}$$

Campos de la Dir.Virtual acc pag(48bits)

Pag. Virtual 32 bits			
Nivel 1	Nivel 2	Nivel 3	Offset
4	14 bits	14 bits	16 bits

Campos del TLB (64 bits)

Etiqueta	MPR	Control
30 bits	16 bits	16 bits

Campos de la Dir.Virtual. acc TLB(48bits)

Pag. Virtual 32 bits		
Etiqueta	Indice	Offset
30 bits	2 bits	16 bits

Dirección Real (32 bits)

M.P.R	Offset
16 bits	16 bits

Campo de la tabla (32 bits)

M.P.R	control
16 bits	16 bits

b) Tamaño del TLB:  $128 \times 62 \text{ bits} = 7936 \text{ bits} = 992 \text{ Bytes}$

La tabla de sustitución directa posee  $2^4 = 16$  posiciones de 32 bits = 512 bits = 64 bytes.

c) Tablas del N2:  $2^4 \times 2^{14} \times 2^2 = 16$  tablas de 16.384 entradas de 4 Bytes =  $2^{20}$  Bytes

Tablas del N3:  $2^{18} \times 2^{14} \times 2^2 = 262.144$  tablas de 16.384 entradas de 4 Bytes =  $2^{34}$  Bytes

Total =  $2^{34} + 2^{20} = 16 \text{ GB} + 1 \text{ MB} = 16.385 \text{ MB}$

Representa aproximadamente el  $2^{34} / 2^{32} \times 100 = 400\%$  Es decir supera en un factor 4 la capacidad de la memoria real.

d) Se necesita un mínimo de 3 accesos a memoria principal:

La tabla de nivel 1 está presente en la MMU y no representa acceso a memoria. Luego un acceso a tabla de nivel 2, otro a la tabla del nivel 3 y finalmente el acceso al dato.