



La carga de una pila o batería se mide en Amperios x hora (Ah), que indica la cantidad de corriente que puede entregar de forma continua durante una hora antes de quedar descargada totalmente. Nuestro procesador se alimenta con una batería de 8Ah a 1,25 V. El procesador consume 5 J durante la ejecución del programa P.

d) **Calcula** la potencia media consumida por nuestro procesador durante la ejecución de P

$$P = E/t = 5J / 2,5s = 2W$$

e) **Calcula** la duración de la batería si el procesador ejecuta el programa P de forma continua?

$$P = I \times V \rightarrow I = 2W / 1,25V = 1,6A$$

$$\text{Duración} = 8Ah / 1,6A = 5 \text{ horas}$$

COGNOMS:

NOM:  DNI/NIE:

**Problema 2. (3.5 puntos)**

Considera el siguiente código escrito en C, suponiendo que las constantes A, B y C han sido declaradas previamente en un #define.

```
typedef struct {
    int t2[B];
    int y;
    int t3[C];
} sa;

typedef struct {
    sa tabla[A];
    int x;
} sb;

void examen(sb *p1, int val)
{
    int idc;

    idc = p1->x;
    p1->tabla[idc].y = val;
}
```

- a) **Dibuja** como quedarían almacenadas en memoria las estructuras sa y sb, indicando claramente los desplazamientos respecto del inicio y el tamaño de todos los campos. Escribir esos valores en función de A, B y C.

----- <--- 0	----- <--- 0
t2[0]	tabla[0]
----- <--- 4	----- <--- tamaño de sa (= 4*(B+1+C))
...	...
-----	-----
t2[B-1]	tabla[A-1]
----- <--- 4*B	----- <--- A*tamaño de sa (= 4*A*(B+1+C))
y	x
----- <--- 4*(B+1)	----- <--- (A*tamaño de sa) + 4
t3[0]	
----- <--- 4*(B+2)	
...	
-----	
t3[C-1]	
----- <--- 4*(B+1+C)	

Tamaño de sb = (A\*tamaño de sa) + 4

= 4\*[A\*(B+1+C) + 1]

Alineamiento a 4 bytes

tamaño de sa = 4\*(B+1+C)

Alineamiento a 4 bytes

- b) **Dibuja** el Bloque de Activación de la función examen, indicando claramente los desplazamientos relativos al EBP necesarios para acceder a los parámetros y las variables locales.

```

|           |
|-----| <--- ebp - 4
|   idc   |
|-----| <--- ebp
|   ebp   |
|-----|
|   @ret  |
|-----| <--- ebp + 8
|   @p1   |
|-----| <--- ebp + 12
|   val   |
|-----|
|           |

```

- c) **Determina** los valores de las constates A, B y C sabiendo que el código en ensamblador del x86 de las dos instrucciones que hay dentro de la función examen es el siguiente. Suponer que en el registro %edi se encuentra 8(%ebp) y en %esi se encuentra val.

**examen:**

```

...
movl 168(%edi), %eax
leal (%eax, %eax, 2), %eax
movl %esi, 12(%edi, %eax, 8)
...

```

A = 7

B = 3

C = 2

COGNOMS:

NOM:  DNI/NIE:

**Problema 3. (3.5 puntos)**

Un procesador consta de una memoria cache con escritura inmediata (write through) y sin carga en caso de fallo de escritura. Mediante contadores hardware se han obtenido las siguientes medidas para un conjunto de programas de referencia:

- porcentaje de escrituras: 20%
- porcentaje de bloques modificados al substituir una línea: 33.33%
- tasa de aciertos 0.9

El tiempo de acceso a memoria cache es de 5 ns y el tiempo de memoria principal para escribir una palabra es de 70 ns. Para leer o escribir un bloque en la memoria principal se emplean 90 ns.

a) **Calcula** el tiempo invertido en realizar 1000 accesos a memoria.

1000 accesos	200 escrituras	hit+miss	70ns	14000 ns
	800 lecturas	720 hits	5ns	3600 ns
		80 miss	100 ns	8000 ns
				25600 ns

La empresa A.C.M.E está diseñando un procesador que sabemos que genera 1.3 referencias a memoria por instrucción de las cuales 0.3 son de datos. Sus diseñadores se han dado cuenta de que en el mismo chip queda espacio suficiente para poner un poco de cache. Después de hacer algunos cálculos han optado por poner dos caches diferenciadas con la siguientes características:

- Dos caches separadas, una de instrucciones de 4Kb y una de datos de 8Kb. El CPI ideal es de 1.2 ciclos/instrucción ya que podemos buscar datos e instrucciones simultáneamente.
- Tc (tiempo de ciclo): 10ns
- Tsa (tiempo de servicio en caso de acierto): 1ciclo.
- Tsf (tiempo de servicio en caso de fallo): 10 ciclos.

La tasa de fallos para la cache de instrucciones es del 8.6% y para la cache de datos es del 6.8%.

b) **Calcula** el tiempo medio de ejecución de 1 instrucción

$\text{CPI Real} = \text{CPI Ideal} + 1 * 9 * 0.086 + 0.3 * 9 * 0.068 = 2,1576$
$\text{Texe} = \text{CPI Real} * Tc = 21,576 \text{ ns}$

Dado el siguiente código escrito en ensamblador x86:

```
    movl $0, %ebx
    movl $0, %esi
for: cmpl $2048*1000, %esi
    jge end
    movw (%ebx, %esi, 2), %ax
    addw %ax, 16*1024(%ebx, %esi, 2)
    movw %ax, 32*1024(%ebx, %esi, 2)
    addl $2048, %esi
    jmp for
end:
```

La memoria utiliza páginas de tamaño 8KB y que utilizamos un TLB de 4 entradas (reemplazo LRU).

c) **Calcula** el porcentaje de aciertos de TLB en todo el bucle.

Las páginas tienen tamaño 8KB y cada iteración del bucle incrementa el acceso en 4KB así que cada 2 iteraciones se accede a una página distinta. Por otro lado en la iteración 0, la primera instrucción accede a la página 0, la segunda a la página 2 y la tercera a la página 4, en la iteración 2 acceden a las páginas 1, 3 y 5 respectivamente de forma que no se pueden reutilizar accesos. Por lo tanto tendremos 3 fallos cada 2 iteraciones u 8 accesos.

Aciertos =  $5/8 \Rightarrow 62,5\%$