Problema 2. Repaso Cache

Disponemos de un procesador de 16 bits con un bus de direcciones de 16 bits. Este procesador tiene una memoria cache de datos con las siguientes características:

- Tamaño de bloque = 16 bytes
- Asociatividad = 2 (reemplazo = LRU)
- Número de líneas = 8
- Política de escritura: write through + write no allocate

Suponiendo que el contenido de la cache es el siguiente:

conjunto 0		conjunto 1		conjunto 2		conjunto 3	
EC8	1	EC5	1	EC6	0	EC7	1
AB4	0	libre	0	AB2	1	libre	0

Teniendo en cuenta que:

- En el contenido de la MC para simplificar hemos dejado el número de bloque de memoria en vez del tag.
- El bit a 1, en el contenido de la cache, indica que es la línea más recientemente referenciada.
- R-byte (lectura de 1 byte), R-word (lectura de 2 bytes), W-byte (escritura de 1 byte), W-word (escritura de 2 bytes).
- El tamaño de las lecturas (y escrituras) se ha de indicar en bytes.

Rellenad la siguiente tabla:

Tipo	@ en hex	Bloqu de memoria	Conjunto de MC	Acierto / Fallo	Lectura de MP			Escritura en MP		
					si / no	@	tamaño	si / no	@	tamaño
R byte	8890									
W word	EC51									
W byte	EC62									
W word	23D3									
W byte	ABA4									
R word	ABA5									
R byte	23D6									
W word	EC57									
R byte	EC68									
R word	8899									

Indicad también cómo queda la cache después de realizar los 10 accesos a memoria:

conjunto 0		conjunto 1		conjunto 2		conjunto 3	

Problema 10. Predicción de vía

Una CPU funciona a un voltaje de 1,2 V y una frecuencia de 2GHz. Se ha determinado que esta CPU tiene una corriente de fugas de 3 A y que a pleno rendimiento tiene una carga capacitiva equivalente de 5 nF (nanoFaradios).

a) **Calculad** la potencia media dinámica (debida a conmutación), la potencia media estática (debida a fugas) y la potencia media total.

Se desea integrar esta CPU con una memoria cache de datos 2-asociativa de 128 KB de capacidad y un tamaño de bloque de cache de 64 bytes. Las direcciones generadas por la CPU son de 48 bits. La corriente de fugas de la memoria RAM estática es de 3 µA (microAmperios) por bit. La energía consumida durante un acceso a la memoria de etiquetas es de 5 nJ (nanoJoules) por vía y la consumida durante un acceso a la memoria de datos es de 25 nJ por vía.

b) **Calculad** el numero de conjuntos, el de bloques de cache, el de vías y el de bloques por vía.
c) **Dibujad** una dirección indicando claramente los campos usados para seleccionar el byte dentro del bloque, seleccionar el conjunto de la cache y los bits usados como etiqueta.
d) **Calculad** el tamaño **en bits** de la memoria de datos y el de la memoria de etiquetas de una vía (por simplicidad ignoraremos el bit de validez y otros bits de control).
e) **Calculad** la potencia media estática (debida a fugas) de la cache.

Se desean comparar diversas implementaciones alternativas de cache de datos 2-asociativa: **paralela**, **serie**, y con **predictor de vía**. Para compararlas se usa un *benchmark* con $4x10^9$ instrucciones dinámicas que realiza 10^9 accesos de datos a memoria y $2x10^9$ operaciones aritméticas de punto flotante. Este *benchmark* tiene un 10% de fallos en la cache descrita anteriormente.

En la implementación **paralela**, se accede simultáneamente tanto a las memorias de etiquetas como las de datos de ambas vías. Un acceso a cache se realiza en 1 ciclo y la penalización media por fallo de cache es de 20 ciclos. El *benchmark* ejecutado con la implementación **paralela** de la cache ha tardado 5 segundos.

f) **Calculad** los MFLOPS de la implementación **paralela.**
g) **Calculad** el CPI de la implementación **paralela** y el CPI que obtendríamos con una memoria ideal (CPI_{ideal}) en donde todos los accesos tardan 1 ciclo.
h) **Calculad** la energía dinámica consumida por un acceso a la cache. Para simplificar asumiremos que todos los accesos consumen lo mismo sean acierto o fallo, la energía extra consumida en acceder a memoria principal en caso de fallo está fuera de los objetivos de este problema.
i) **Calculad** la potencia (dinámica) media consumida en acceder a la cache
j) **Calculad** la potencia media total (estática+dinámica) consumida por el sistema CPU-cache.
k) **Calculad** la energía total consumida para ejecutar el *benchmark* y la eficiencia en MFLOPS/Watt.

En la implementación **serie** un acceso tarda 2 ciclos. En el primer ciclo se accede a las memorias de etiquetas de ambas vías. Una vez determinada la vía que contiene el dato, en el segundo ciclo se accede solo a la memoria de datos de dicha vía. La penalización en caso de fallo sigue siendo de 20 ciclos ya que en el primer ciclo del acceso ya se puede determinar si es acierto o fallo. Obsérvese que respecto la implementación **paralela**, los aciertos tienen una penalización de 1 ciclo.

l) **Calculad** el tiempo de ejecución y los MFLOPS de la implementación **serie.**
m) **Calculad** la energía consumida por un acceso a la cache.

n) **Calculad** la potencia (dinámica) media consumida en acceder a la cache
o) **Calculad** la potencia media total consumida por el sistema CPU-cache.
p) **Calculad** la energía total consumida para ejecutar el *benchmark* y la eficiencia en MFLOPS/Watt.

En la implementación con **predictor de vía**, este predice la vía probable en que se encuentra el dato y se accede solo a las memorias de etiquetas y de datos de esa vía. En caso de fallo del predictor hay que acceder a la otra vía (memorias de etiquetas y datos). El predictor usado consiste en una memoria de 8k x 1 bits indexado con los bits bajos del PC de las instrucciones de acceso a memoria. Este predictor tiene una tasa de aciertos del 80% del total de accesos a memoria y cada acceso al predictor consume 1nJ. En esta implementación se pueden dar las siguientes situaciones:

- El predictor acierta: se accede a 1 sola vía (datos+etiquetas) en 1 ciclo (no hay penalización) y al comprobar los tags se comprueba que es acierto de cache.
- El predictor falla pero es acierto de cache: El acceso tarda 2 ciclos, en el primero se accede a la vía probable (aunque equivocada) al comprobar los tags se descubre que es fallo (de via) y en el segundo se accede a la vía correcta y se descubre que es acierto de cache (1 ciclo de penalización).
- El predictor falla y además es fallo de cache: En el primer ciclo se accede a la vía probable (aunque incorrecta), en el segundo ciclo se accede a la otra vía y se descubre que es fallo de cache (con lo que hay que acceder a memoria principal). Obsérvese que en este caso la penalización es de 21 ciclos ya que no se descubre el fallo de cache hasta que se accede a la 2a vía.

q) ¿Puede darse al caso de que un acierto del predictor de vía sea fallo de cache? ¿porqué?
r) **Calculad** la potencia media estática (debida a fugas) del predictor y compárala con la de la cache (se calcula de la misma forma ya que se ha empleado el mismo tipo de memoria estática).
s) **Calculad** el tiempo de ejecución y los MFLOPS de la implementación con **predictor de vía.**
t) **Calculad** la energía consumida por un acceso en que el predictor acierta y uno en que el predictor falla (tener en cuenta la energía consumida por el acceso al predictor). Calcular también la energía media consumida por acceso.
u) **Calculad** la potencia (dinámica) media consumida en acceder a la cache
v) **Calculad** la potencia media total consumida por el sistema CPU-cache (acuérdate de las fugas del predictor).
w) **Calculad** la energía total consumida para ejecutar el *benchmark* y la eficiencia en MFLOPS/Watt.
x) **Calculad** la ganancia en eficiencia energética de la implementación serie sobre la paralela y la de predicción de vía sobre la serie.

Problema 11. Caches segmentadas

En un procesador X el camino crítico, y por tanto el tiempo de ciclo, está limitado por la memoria cache de datos. El tiempo de los componentes de la memoria cache de datos se desglosa de la siguiente forma:

Componente	Tiempo
Memoria de etiquetas	0,30 ns
Selección de vía	0,15 ns
Memoria de datos	0,45 ns
Mux/Driver de datos	0,10 ns
Registro de desacoplo (en caso que se use)	0,05 ns

Queremos evaluar el rendimiento de 4 posibles implementaciones de la memoria cache para el procesador X:

- Procesador **X1**: La cache de datos tiene una implementación paralela y el tiempo de acceso a la cache es de 1 ciclo de procesador
- Procesador **X2**: La cache de datos está segmentada en 2 etapas y el tiempo de acceso a la cache es de 2 ciclos de procesador
- Procesador **X3**: La cache de datos está segmentada en 3 etapas y el tiempo de acceso a la cache es de 3 ciclos de procesador
- Procesador **X4**: La cache de datos está segmentada en 4 etapas y el tiempo de acceso a la cache es de 4 ciclos de procesador

a) **Calculad** el tiempo de ciclo de la cache de datos y el tiempo total de un acceso para los procesadores X1, X2, X3 y X4, usando la distribución mas adecuada de los componentes por etapas.

b) **Razonad** porque descartamos las opciones X2 y X4 para el resto del problema

c) **Calculad** la frecuencia de reloj de los procesadores X1 y X3

Un programa P que ejecuta $2x10^9$ instrucciones tiene un 60% de instrucciones aritméticas, un 20% de instrucciones de salto y un 20% de instrucciones de acceso a memoria (Load/Store). Las instrucciones aritméticas tardan 5 ciclos, las de salto 4 y las de memoria 4 ciclos + los ciclos del acceso a la cache.

d) **Calculad** el CPI del programa P para los procesadores X1 y X3 suponiendo que nunca hay fallos en la cache de datos.

e) **Calculad** el speedup de X3 sobre X1 en % suponiendo que nunca hay fallos en la cache de datos

Sabemos que el programa P tiene un 10% de fallos en la cache de datos utilizada y que el tiempo de penalización en ambos casos es de 60 ciclos.

f) **Calculad** el speedup real de X3 sobre X1 en % teniendo en cuenta la jerarquía de memoria completa.