Problema 18. Cache Multinivell, DRAM

Tenim el disseny de una CPU que tindrà un temps de cicle (Tc) de 10 ns. A l'executar un programa P (que executa $5x10^9$ instruccions) en un simulador on tots els accessos a memòria tarden 1 cicle s'ha mesurat un CPI de 1,8 cicles/instrucció (que anomenarem CPI_{ideal}).

a) **Calculeu** el temps d'execució (T_{exec}) del programa P en aquest sistema de memòria ideal (en segons).

Per mesurar l'impacte de la cerca d'instruccions en el rendiment, hem modificat el simulador per analitzar un sistema amb una cache d'instruccions (que anomenarem L1) i una memòria principal SDRAM (els accessos a dades segueixen tardant 1 cicle). La mida de bloc de L1 es de 32 bytes i el temps d'accés en cas d'encert a L1 (T_h) es de 1 cicle. Pel programa P la taxa de fallades (m_1) de L1 es del 10%. La memòria principal esta formada per un DIMM SDRAM de 8 bytes d'amplada amb una latència de fila de 4 cicles, una latència de columna també de 4 cicles i un temps per la comanda PRECHARGE de 1 cicle.

b) **Calculeu** quants accessos a L1 fa el programa P.

El següent cronograma mostra els passos en cas de fallada a L1. En el cicle 1 s'accedeix a L1 i es detecta que es una fallada de cache. En el cicle 2 s'envia la comanda ACTIVE i l'adreça de fila (Bus A) per activar la pàgina corresponent de memòria i 4 cicles després (cicle 6) s'envia la comanda READ i l'adreça de columna. Al cicle 10 (4 cicles després de RD) apareixen les dades (4 cicles 8 bytes/cicle) al bus de dades (Bus D). Les dades es van carregant a un *buffer* (cicles CB) a mesura que van apareixent pel bus (cicles etiquetats D). Finalment al cicle 14, un cop s'ha transmès tot el bloc al *buffer*, es passa la instrucció a la CPU (DADA) i en paral.lel s'escriu el bloc a L1 (carL1) i s'activa la comanda PRECHARGE per tancar la pàgina (PRE).

CLK																
Cicle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
CPU														DADA		
L1	MISS													carL1		
Buffer										CB	CB	CB	CB			
Com		ACT				RD								PRE		
Bus A		@F				@C										
Bus D										D	D	D	D			

c) **Calculeu** el temps de penalització d'una fallada (en cicles).

d) **Calculeu** el temps mig d'accés a memòria (T_{mam}) pels accesos a instruccions (en nanosegons).

e) **Calculeu** el CPI amb aquesta jerarquia de memòria.

f) **Calculeu** el temps d'execució (T_{exec}) del programa P (en segons).

A aquest sistema afegim un segon nivell de cache (L2) entre la cache d'instruccions (L1) i la memòria principal (SDRAM) de forma que, si es falla a L1 s'accedeix a L2 i nomes en cas de fallar al segon nivell s'ha d'accedir a memòria principal. La taxa local de fallades (m_2) de L2 es del 30%. La mida de bloc de L2 es també de 32 bytes.

g) **Calculeu** el percentatge d'accessos que fallen a L1 i encerten a L2.

h) **Calculeu** el percentatge d'accessos que fallen a L1 i a L2.

El següent cronograma mostra els passos en cas de fallada a L1 i encert a L2. Al cicle 1 s'accedeix a L1 i es detecta que es una fallada a L1. Un accés a L2 tarda 4 cicles (del 2 al 5). En el cicle 2 (TAG) es llegeix la memòria d'etiquetes, en el cicle 3 (CMP) es comparen les etiquetes i es comprova que es *hit* a L2, en el cicles 4 i 5 es llegeix la memòria de dades de la L2 (RD1 i RD2). Finalment al cicle 6 s'escriu el bloc a L1 (carL1) i, en paral.lel, es passa la instrucció a la CPU (DADA).

CLK																
Cicle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
CPU						DADA										
L1	MISS					carL1										
L2		TAG	CMP	RD1	RD2											

i) **Calculeu** el temps de penalització en cas de fallar a L1 i encertar a L2 (en cicles).

El següent cronograma mostra els passos en cas de fallada a L1 i a L2. Al cicle 1 s'accedeix a L1 i es detecta que es una fallada a L1. En el cicle 2 (TAG) es llegeix la memòria d'etiquetes, en el cicle 3 es comparen les etiquetes i es comprova que es *miss* a L2. Dels cicles 4 al 15 es llegeix el bloc de la SDRAM tal com ja s'ha explicat per la configuració amb un sol nivell de cache. Un cop tenim el bloc al *buffer*, aquest s'escriu simultaniament a L1 (carL1), L2 (2 cicles WR1 i WR2) i es passa la instrucció a la CPU (DADA).

CLK																	
Cicle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
CPU																DADA	
L1	MISS															carL1	
L2		TAG	CMP													WR1	WR2
Buffer												CB	CB	CB	CB		
Com				ACT				RD								PRE	
@				@F				@C									
Datos												D	D	D	D		

j) **Calculeu** el temps de penalització en cas de fallar a L1 i a L2 (en cicles).

k) **Calculeu** el temps mig d'accés a memòria (T_{mam}) pels accesos a instruccions (en nanosegons).

l) **Calculeu** el CPI amb aquesta jerarquia de memòria.

m) **Calculeu** el temps d'execució (T_{exec}) del programa P (en segons).

n) **Calculeu** el guany (*speed-up*) del sistema amb L1 i L2 respecte el sistema que només te L1.

Problema 1. Discos, Ancho de banda, RAIDs

Disponemos de discos con las siguientes características:

- Capacidad 1 Tbyte.
- Seek time medio (situar el cabezal en el cilindro) 8 ms.
- Latencia media (tiempo que tarda en pasar el sector deseado) 2 ms.
- Transfer Rate (ancho de banda durante la transferencia de datos) 256 Mbytes/s.
- MTTF 50.000 horas.
- Tamaño de sector 512 bytes.

a) **Calcula** cuanto tiempo tarda en transferirse un bloque de datos de 5000 sectores.

Si los 5000 sectores se encuentran almacenados de forma consecutiva en la misma pista, solo se pierde tiempo en situar el cabezal (seek time + latencia) en el primero de ellos ya que el disco es capaz de transferir los siguientes sectores a medida que gira el disco sin necesidad de situar el cabezal (suponemos que la velocidad de rotación es la adecuada para al Transfer Rate).

b) **Calcula** el tiempo total necesario para leer un bloque de datos de 5000 sectores consecutivos desde que se envía la petición al disco hasta que los datos están en memoria (suponemos que el procesado de las peticiones por el disco, calculo de CRCs, DMA, etc introducen un tiempo negligible).

c) **Calcula** el ancho de banda efectivo al leer un bloque de datos de 5000 sectores consecutivos.

En un computador con uno de estos discos ejecutamos una aplicación formada por 3 fases:

- La fase 1 lee 8 bloques de datos (de 5000 sectores consecutivos cada uno) de disco.
- La fase 2 realiza los calculos y representa el 40% del tiempo de la aplicación (ejecutada con un solo disco).
- La fase 3 escribe 4 bloques de datos (de 5000 sectores consecutivos cada uno) a disco. Escribir un bloque de datos tarda lo mismo que leerlo.

d) **Calcula** el tiempo que tarda la fase 2.

Aunque los datos caben en un solo disco, para mejorar el rendimiento de la aplicación, instalamos un RAID 0 con 8 discos iguales con tiras de 5000 sectores de forma que los 8 bloques de datos de la fase 1 se encuentran en 8 discos distintos. Igualmente, los 4 bloques de la fase 3, también se encuentran en 4 discos distintos. Suponemos que el resto del sistema es capaz de soportar el ancho de banda necesario.

e) **Calcula** el ancho de banda efectivo al leer los 8 bloques de datos del RAID 0

f) **Calcula** el ancho de banda efectivo al escribir los 4 bloques de datos al RAID 0

g) **Calcula** el speed-up de la fase 1

h) **Calcula** el speed-up de la fase 3

i) **Calcula** el speed-up de la aplicación

Problema 2. RAIDS, ancho de banda

Disponemos de 60 discos físicos de 300 Gbytes de capacidad por disco, que ofrecen un ancho de banda efectivo de 100 Mbytes/s por disco. Con estos discos deseamos montar un disco lógico en donde consideramos las siguientes 4 opciones:

- RAID 6
- RAID 10 (mirror doble con 30 grupos de 2 discos)
- RAID 50 (con 6 grupos de 10 discos)
- RAID 51 (mirror doble con 2 grupos de 30 discos)

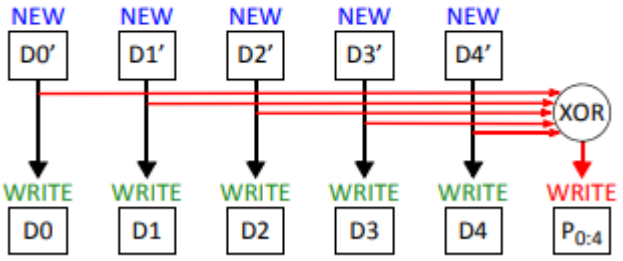
a) **Calcular** la cantidad de información útil que puede almacenar cada uno de los RAIDS considerados.

Para analizar el ancho de banda consideraremos por separado el caso en que realizamos accesos a tiras consecutivas y el caso en que realizamos accesos a tiras aleatorias. En el caso de accesos aleatorios, el controlador del RAID ordena las peticiones de acceso para aprovechar al máximo la concurrencia entre discos. En ambos casos consideraremos que disponemos de suficientes peticiones de lectura de forma que el controlador del RAID siempre puede aprovechar el ancho de banda de todos los discos físicos.

b) **Calcular** el ancho de banda efectivo, si hacemos **lecturas** secuenciales, para cada uno de los RAIDS considerados.

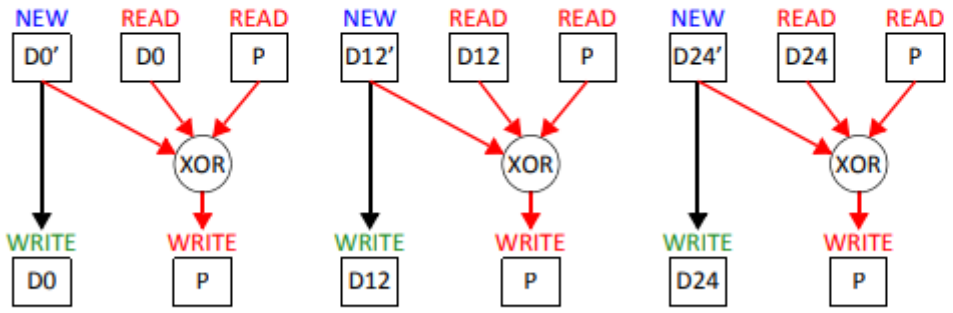
c) **Calcular** el ancho de banda efectivo, si hacemos **lecturas** aleatorias, para cada uno de los RAIDS considerados.

Para aquellos RAIDS que tienen algún tipo de paridad (como 6, 50 o 51), en el caso de escrituras secuenciales es posible esperar a tener una cantidad de datos suficiente para calcular la paridad correspondiente y escribir simultáneamente en todos los discos. La siguiente figura muestra la escritura de 5 tiras de datos consecutivas D0-D4 en lo que podria ser un RAID 4 o 5 con 6 discos. Obsérvese que podemos aprovechar el ancho de banda de 5 discos para datos ya que el sexto es usado para escribir la paridad.



d) **Calcular** el ancho de banda efectivo, si hacemos **escrituras** secuenciales, para cada uno de los RAIDS considerados.

En el caso de escrituras aleatorias (para RAIDS con algún tipo de paridad), se ha visto en teoría que es necesario leer los datos antiguos y la tira (o tiras) de paridad correspondiente para calcular la nueva paridad y además escribir tanto datos como paridad por cada escritura que se desea realizar. La siguiente figura muestra la escritura en paralelo de 3 tiras de datos independientes en lo que podría ser un RAID 5 con 6 discos (para acabarlo de entender es recomendable dibujar como estarían distribuidas las tiras de la 0 a la 24 y las paridades correspondientes en un RAID 5 con 6 discos). Obsérvese que para escribir 3 tiras de datos es necesario realizar 12 operaciones de disco realizando una lectura y una escritura en cada uno de los 6 discos, con lo que en la práctica el ancho de banda efectivo (de datos) es equivalente a 1,5 discos.



e) **Calcular** el ancho de banda efectivo, si hacemos **escrituras** aleatorias, para cada uno de los RAIDS considerados.