Arquitectura e Ingeniería de Computadores - 3º Grado Ing. Informática - ETSINF - UPV

Ejercicios de la Unidad Temática 3. "Subsistema de memoria"

Ejercicio 3.1.

En la fase de diseño de la cache de un computador se están considerando dos alternativas. La primera es utilizar una correspondencia directa, lo que da lugar a una tasa de fallos del 1.4 %, fijando el periodo de reloj en 1 ns. La segunda consiste en utilizar una correspondencia asociativa, lo que permite reducir la tasa de fallos al 1 %, pero el periodo de reloj aumenta hasta 1.25 ns. En caso de acierto, se consume un ciclo de reloj en leer o escribir el dato. Si los programas se ejecutan con un CPI de 2, con 1.5 accesos a memoria por instrucción y la penalización por fallo es de 75 ns, decídase qué alternativa es la mejor.

Solución:

El tiempo de acceso a memoria en un sistema con memorias cache es:

$T_{acceso} = TA + TF \times PF$

En nuestro caso:

- Correspondencia directa:
  $T_{acceso} = 1 + 0{,}014 \times 75 = 2{,}05$ ns.
- Correspondencia asociativa:
  $T_{acceso} = 1{,}25 + 0{,}01 \times 75 = 2$ ns.

Resulta mejor la correspondencia asociativa, pues reduce el tiempo de acceso.

Seguidamente, compararemos ambas alternativas desde el punto de vista del tiempo de ejecución. El tiempo de ejecución de un programa en un computador con memoria cache viene dado por:

$T_{ejecucion} = I \times CPI \times T + I \times API \times TF \times PF \times T$. Sustituyendo:

- Correspondencia directa:
  Como el tiempo de acceso a memoria es de 75 ns y el periodo de reloj es de $T = 1$ ns, la penalización por fallo en ciclos es de $PF = 75$:
  $T_{ejecucion} = I \times 2 \times 1 + I \times 1{,}5 \times 0{,}014 \times 75 \times 1 = 3{,}58I$ ns.
- Correspondencia asociativa:
  Como el tiempo de acceso a memoria es de 75 ns y el periodo de reloj es de $T = 1{,}25$ ns, la penalización por fallo en ciclos es de $PF = 60$:
  $T_{ejecucion} = I \times 2 \times 1{,}25 + I \times 1{,}5 \times 0{,}01 \times 60 \times 1{,}25 = 3{,}63I$ ns.

La correspondencia directa, con menor periodo de reloj, reduce el tiempo de ejecución.

□

Ejercicio 3.2.

Analizar el tiempo de ejecución de un programa en un computador segmentado con CPI=1 bajo las siguientes configuraciones de memoria cache:

1. No emplear memoria cache.
2. Utilizar una cache con una tasa de fallos del 2 %.
3. Utilizar una cache perfecta.

La memoria principal tiene 100 ciclos de reloj de tiempo de acceso y el programa tiene un 50 % de instrucciones de acceso a memoria.

Solución:

El tiempo de acceso a memoria en un sistema con memorias cache es:

$T_{acceso} = TA + TF \times PF$

El tiempo de ejecución de un programa en un computador con memoria cache viene dado por:

$T_{ejecucion} = I \times CPI \times T + I \times API \times TF \times PF \times T.$

Sustituyendo:

1. No emplear memoria cache.
   $T_{acceso} = 1 + 1(100\,\%) \times 100 = 101$ ciclos
   $T_{ejecucion} = I \times 1 \times T + I \times 1{,}5 \times 1(100\,\%) \times 100 \times T = 151I$ ciclos.
2. Utilizar una cache con una tasa de fallos del 2 %.
   $T_{acceso} = 1 + 0{,}02 \times 100 = 3$ ciclos
   $T_{ejecucion} = I \times 1 \times T + I \times 1{,}5 \times 0{,}02 \times 100 \times T = 4I$ ciclos.
3. Utilizar una cache perfecta.
   $T_{acceso} = 1 + 0{,}0 \times 100 = 1$ ciclos
   $T_{ejecucion} = I \times 1 \times T + I \times 1{,}5 \times 0{,}0 \times 100 \times T = I$ ciclos.

□

Ejercicio 3.3.

Se pretende comparar las prestaciones de dos subsistemas de memoria. El primero tiene sendas memorias cache de instrucciones y datos de 16 KB cada una, mientras que el segundo tiene una única memoria cache de 32 KB. Tras haber ejecutado unos programas de prueba, se ha obtenido que el número de fallos por cada 1000 instrucciones ejecutadas es 3.83 para la cache de instrucciones, 40.9 para la cache de datos y 43.3 para la cache unificada, y que el 36 % de las instrucciones ejecutadas son de acceso a memoria. Si el tiempo de acierto y de fallo es de 1 y 100 ciclos de reloj, respectivamente:

1. Calcular el tiempo de acceso a memoria en ambos esquemas.
2. Si el computador está segmentado y el programa de prueba se compone de n instrucciones, comparar las dos alternativas en base al tiempo de ejecución.

Solución:

1. Calculo del tiempo de acceso a memoria en ambos esquemas.
   El tiempo de acceso a memoria en un sistema con memorias cache es:

   $$T_{acceso} = TA + TF \times PF$$

   En nuestro caso:
   TF (fallos/acceso) = (fallos/instruccion)/(accesos/instruccion). Sustituyendo:
   $TF_I = \frac{3{,}82/1000}{1} = 0{,}004$ $TF_D = \frac{40{,}9/1000}{0{,}36} = 0{,}1144$ $TF_U = \frac{43{,}3/1000}{1{,}36} = 0{,}0318$
   Caches separadas:
   Un programa compuesto por 100 instrucciones realiza por término medio 36 accesos a memoria de datos, con 136 accesos en total. Por lo tanto, la tasa de fallos vendrá dada por:
   $TF_{I+D} = \frac{100}{136}TF_I + \frac{36}{136}TF_D = 0{,}74 \cdot 0{,}004 + 0{,}26 \cdot 0{,}1144 = 0{,}0324$. Por lo tanto;
   $T_{acceso} = TA + TF \times PF = 1 + 0{,}0324 \cdot 100 = 4{,}24$ ciclos
   Cache unificada:
   En este caso, hay que tener presente que los accesos a memoria de datos (26 % del total de accesos) van a ocasionar un riesgo estructural que se saldará con un ciclo de parada:
   $T_{acceso} = TA + TF \times PF = 1 + 0{,}26 \cdot 1 + 0{,}0318 \cdot 100 = 4{,}44$ ciclos

2. Comparación de las dos alternativas en base al tiempo de ejecución.
   El tiempo de ejecución de un programa en un computador con memoria cache viene dado por:
   $T_{ejecucion} = I \times CPI \times T + I \times API \times TF \times PF \times T$. Sustituyendo:
   Caches separadas:
   $T_{ejecucion} = n \times 1 \times T + n \times 1{,}36 \times 0{,}0324 \times 100 \times T = 5{,}41n$ ciclos.
   Cache unificada:
   $T_{ejecucion} = n \times (1 + 0{,}36) \times T + n \times 1{,}36 \times 0{,}0318 \times 100 \times T = 5{,}49n$ ciclos.

□

Ejercicio 3.4.

En el diseño de un computador segmentado con CPI=1 se pretende evaluar dos alternativas para el sistema de antememorias. La primera basada en una cache de dos niveles. La segunda formada por una cache asociativa por conjuntos de cuatro vías. A continuación se detallan los datos técnicos de ambas alternativas.

Cache multinivel Está compuesta por dos niveles de cache.
Primer nivel: directa Tiempo de acierto, $TA_{L1} = 1$ ciclo; tasa de fallos local, $TF_{L1} = 0{,}04$.
Segundo nivel: asociativa Tiempo de acierto, $TA_{L2} = 5$ ciclos; tasa de fallos local, $TF_{L2} = 0{,}50$.

Cache asociativa Una cache asociativa por conjuntos de cuatro vias.
Tiempo de acierto, $TA = 1$ ciclo; tasa de fallos local, $TF = 0{,}02$.

En ambos casos, la memoria principal tiene un tiempo de acceso de 50 ciclos. También se sabe que el 25 % de las instrucciones ejecutadas son de carga o almacenamiento.
Se solicita:

1. Calcular el tiempo de acceso a memoria en ambos esquemas.
2. Comparar las dos alternativas en base al tiempo de ejecución.

Solución:

Cache multinivel:

El tiempo de acceso de una cache multinivel es:

$$T_{acceso} = TA_{L1} + TF_{L1} \times \underbrace{(TA_{L2} + TF_{L2} \times PF_{L2})}_{PF_{L1}}$$

sustituyendo:

$$T_{acceso} = 1 + 0{,}04 \times \underbrace{(5 + 0{,}5 \times 50)}_{PF_{L1}} = 1 + 0{,}04 \times \underbrace{30}_{PF_{L1}} = 2,2 \text{ ciclos}$$

Utilizando el tiempo de PF_{L1}, el tiempo de ejecución sería:

$$T_{ejecucion} = I \times CPI \times T + I \times API \times TF \times PF \times T$$

$$T_{ejecucion} = I \times (CPI + API \times TF \times PF) \times T$$

sustituyendo:

$$T_{ejecucion} = I \times (1 + 1{,}25 \times 0{,}04 \times 30) \times T = I \times 2{,}5 \times T$$

Cache asociativa:

El tiempo de acceso de una cache convencional es:

$$T_{acceso} = TA_{L1} + TF_{L1} \times PF_{L1}$$

sustituyendo:

$$T_{acceso} = 1 + 0{,}02 \times 50 = 2 \text{ ciclos}$$

Por otro lado, el tiempo de ejecución sería:

$$T_{ejecucion} = I \times (CPI + API \times TF \times PF) \times T$$

sustituyendo:

$$T_{ejecucion} = I \times (1 + 1{,}25 \times 0{,}02 \times 50) \times T = I \times 2{,}25 \times T$$

Así pues, la segunda alternativa es mejor con una aceleración:

$$S = \frac{T_{multinivel}}{T_{asociativa}} = \frac{I \times 2{,}5 \times T}{I \times 2{,}25 \times T} = 1{,}11$$

□

Ejercicio 3.5.

En el diseño de un computador segmentado ideal se dispone de una jerarquía de memoria compuesta por un sistema de antememorias de dos niveles y una memoria principal. Las características son las siguientes:

Primer nivel: correspondencia directa Tiempo de acierto, $TA_{L1} = 1$ ciclo; tasa de fallos local, $TF_{L1} = 0{,}04$.

Segundo nivel: correspondencia asociativa Tiempo de acierto, $TA_{L2} = 5$ ciclos; tasa de fallos local, $TF_{L2} = 0{,}50$.

Memoria principal Tiempo de acceso, $TA_{MP} = 20$ ciclos.

Se sabe que el 25 % de las instrucciones ejecutadas son de carga o almacenamiento.

Se propone mejorar el acceso a la memoria principal utilizando la técnica de *palabra crítica cuanto antes* (*critical word first*), que reduce el tiempo de acceso efectivo a la memoria principal a $TA_{MP} = 15$ ciclos. Introducir esta mejora implica aumentar el periodo de reloj en un 10 %.

El aumento en el periodo de reloj nos permite aumentar la asociatividad del segundo nivel de antememorias, reduciendo la tasa de fallos local a $TF_{L2} = 0{,}40$.

Se solicita cuantificar la mejora obtenida con respecto al tiempo de ejecución.

Solución:

1. El tiempo de acceso de la cache multinivel original es:

$$T_{acceso} = TA_{L1} + TF_{L1} \times \underbrace{(TA_{L2} + TF_{L2} \times PF_{L2})}_{PF_{L1}}$$

$$PF_{L1} = TA_{L2} + TF_{L2} \times PF_{L2} = 5 + 0{,}5 \times 20 = 5 + 10 = 15$$

El tiempo de ejecución sería:

$$T_{ejecucion} = I \times CPI \times T + I \times API \times TF \times PF \times T$$

$$T_{ejecucion} = I \times (CPI + API \times TF \times PF) \times T$$

sustituyendo:

$$T_{ejecucion} = I \times (1 + 1{,}25 \times 0{,}04 \times 15) \times T = I \times 1{,}75 \times T$$

La penalización por fallo de nivel L1 con la mejora indicada sería:

$$PF'_{L1} = TA'_{L2} + TF'_{L2} \times PF'_{L2} = 5 + 0{,}4 \times 15 = 5 + 6 = 11$$

Utilizando este tiempo de PF_{L1}, el tiempo de ejecución sería:

$$T'_{ejecucion} = I' \times (CPI' + API' \times TF' \times PF') \times T'$$

sustituyendo:

$$T'_{ejecucion} = I \times (1 + 1{,}25 \times 0{,}04 \times 11) \times 1{,}1 \cdot T$$
$$T'_{ejecucion} = I \times 1{,}55 \times 1{,}1 \cdot T = I \times 1{,}705 \times T$$

Así pues, la mejora obtenida sería:

$$S = \frac{T_{ejecucion}}{T'_{ejecucion}} = \frac{I \times 1{,}75 \times T}{I \times 1{,}705 \times T} = 1{,}026$$

El tiempo de ejecución mejoraría en un $2,6\,\%$.

□

Ejercicio 3.6. Se está diseñando un computador segmentado de 32 bits con antememorias de instrucciones y de datos separadas. Las características del esquema de memoria son:

- Memoria principal: latencia, 16 $ciclos$; ancho de banda, $W = 2\ bytes/ciclo$. Con lo que, por ejemplo, el tiempo de transferencia de un bloque de 6 palabras sería: $T_{mp} = 16 + \frac{(6\times4)}{2} = 28\ ciclos$.
- Antememoria de datos: Tiempo de acierto, $TA^D = 1\ ciclo$; tasa de fallos, $TF^D = 0{,}05$.
- Antememoria de instrucciones: Tiempo de acierto, $TA^I = 1\ ciclo$; tasa de fallos, $TF^I = 0{,}02$.
- Tamaño de bloque, $B = 4\ palabras$.

Se pretenden evaluar dos posibles mejoras para el sistema de antememorias. A continuación se detallan los datos técnicos de ambas alternativas.

1) Incorporar un segundo nivel de cache con: Tiempo de acierto, $TA_{L2} = 6\ ciclos$; tasa de fallos local, $TF_{L2} = 0{,}5$. El tamaño de bloque se mantiene.
2) Aumentar el tamaño de bloque de ambas antememorias a $8\ palabras$, duplicando así su tamaño. Este cambio afectará a la tasa de fallos de ambas antememorias: $TF^D = 0{,}03$ y $TF^I = 0{,}015$

El CPI, ignorando los fallos de antememoria, es de $1{,}1\ ciclos/inst$ y se sabe que el 25 % de las instrucciones ejecutadas son de carga o almacenamiento. Se solicita analizar la influencia de ambas alternativas sobre el tiempo de ejecución de los programas, indicando cuál de las dos alternativas es la más interesante y cuantificando la mejora obtenida sobre el diseño original.

Solución:

El tiempo necesario para traer un bloque de B palabras de la memoria principal viene dado por la expresión:
$T_{mp} = 16 + \frac{(B\times4)}{2} = 16 + B \times 2$ ciclos.
El tiempo de ejecución de teniendo en cuenta la jerarquía de memorias sería:

$$T_{ejec} = I \times CPI \times T + I \times API \times TF \times PF \times T$$

Cuando se tiene una arquitectura con antememoria de datos y de instrucciones el numero de accesos por instrucción y la tasa de fallos se pueden separar:

$$T_{ejec} = I \times CPI \times T + I \times \Big((API^I \times TF^I + API^D \times TF^D) \times PF\Big) \times T$$

Aplicándolo al diseño original del subsistema de memoria:
$CPI = 1{,}1$, $API^I = 1$, $TF^I = 0{,}02$, $API^D = 0{,}25$, $TF^D = 0{,}05$ y $PF = 16 + 4 \times 2 = 24$ ciclos.
Sustituyendo:

$$T_{ejec} = I \times 1{,}1 \times T + I \times ((1 \times 0{,}02 + 0{,}25 \times 0{,}05) \times 24) \times T = I \times 1{,}88 \times T$$

Veamos ahora el tiempo de ejecución con cada una de las mejoras propuestas:

- Dos niveles de cache.
  En este caso, la penalización en caso de fallo queda:
  $PF = TA_{L2} + TF_{L2} \times T_{mp} = 6 + 0{,}5 \times 24 = 18$ ciclos.
  Sustituyendo en la expresión del tiempo de ejecución:

$$T_{ejec} = I \times 1{,}1 \times T + I \times ((1 \times 0{,}02 + 0{,}25 \times 0{,}05) \times 18) \times T = I \times 1{,}69 \times T$$

- Aumentar el tamaño de bloque.
  En este caso, además de las tasas de fallo, se modifica la penalización en caso de fallo, al cambiar el tamaño de bloque:
  $PF = 16 + 8 \times 2 = 32$ ciclos.
  Sustituyendo en la expresión del tiempo de ejecución:

$$T_{ejec} = I \times 1{,}1 \times T + I \times ((1 \times 0{,}015 + 0{,}25 \times 0{,}03) \times 32) \times T = I \times 1{,}82 \times T$$

Claramente, la primera opción es la mejor. La aceleración obtenida respecto del diseño original es $\frac{I \times 1{,}88 \times T}{I \times 1{,}69 \times T} = 1{,}1124$. Por lo tanto, utilizar dos niveles de cache mejora el tiempo de ejecución en un 11,24 %.

□

Ejercicio 3.7.

Un computador tiene los siguientes componentes:

Un procesador *load/store* semejante al MIPS, con un ancho de palabra de 32 bits y que trabaja a una frecuencia de 100 MHz, con un CPI = 1.3. El 20 % de las instrucciones que ejecuta son lecturas de memoria, y el 10 % son escrituras.

La memoria cache es de un nivel, separada en 16 KB para instrucciones y 16 KB para datos. El tamaño de bloque es de 16 bytes y la correspondencia es asociativa por conjuntos de dos. Sigue las políticas *write-through* y *no-write allocate*. La tasa de fallos es de 2 % para las lecturas y 5 % para las escrituras.

Para las escrituras, el procesador dispone de buffers, que consiguen priorizar siempre las lecturas.

La memoria principal ofrece una latencia de lectura de 40 ns. Una vez cumplida esta latencia, suministra las palabras en intervalos de 10 ns. Así, la primera palabra llega a los 50ns, la segunda a los 60ns, y así consecutivamente. Inicialmente, la carga de un bloque en la cache es convencional (esto es, no aplica *Critical Word First*–palabra crítica cuanto antes ni *Early Restart*–continuar cuanto antes)

Calcula:

1. La penalización por fallo de lectura, expresada en ciclos de reloj y en segundos, cuando se aplica carga convencional.
2. La penalización por fallo de lectura, expresada en ciclos de reloj y en segundos, cuando se aplica carga *Critical Word First*.
3. La penalización por fallo de escritura, expresada en ciclos de reloj y en segundos
4. El tiempo de ejecución de un programa de 10 millones de instrucciones, suponiendo que se aplica carga convencional.

Solución:

1. Penalización por fallo de lectura con carga convencional:
   Si el ancho de palabra es de 32 bits y el tamaño de bloque de 16 bytes, las ráfagas para cargar un bloque en la cache son de 4 palabras. El tiempo necesario es $PF = (5 + 1 + 1 + 1)ciclos = 8 \times 10ns = 80ns$
2. Penalización por fallo de lectura con carga *Critical Word First*:
   El tiempo necesario es $PF = 5ciclos = 5 \times 10ns = 50ns$
3. La penalización por fallo de escritura: ninguna, pues con la política *no-write allocate* y los buffers de escritura, el procesador nunca ha de esperar para transferencias con memoria

4. El tiempo de ejecución es $T = I \cdot CPI \cdot t_C + I \cdot LPI \cdot TFL \cdot PFL = 10 \cdot 10^6 \cdot 1{,}3 \cdot 10ns + 10 \cdot 10^6 \cdot 1{,}2 \cdot 0{,}02 \cdot 80ns$

□

Ejercicio 3.8. Un computador tiene un procesador segmentado compatible binario con el MIPS y que además incorpora instrucciones aritméticas que operan con datos en memoria (modelo de ejecución registro-memoria). Trabaja a una frecuencia de 100 MHz y tiene una jerarquía de memoria con antememorias de instrucciones y de datos separadas, cada una con un solo puerto. El 15 % de las instrucciones que ejecuta son de carga, el 5 % son de almacenamiento y el 40 % son instrucciones aritméticas, de las cuales el 25 % tienen un operando en memoria.

Las etapas que atraviesan las instrucciones son: IF (búsqueda de la instrucción), ID (decodificación y lectura de operandos en registros), ME1 (lectura de los operandos en memoria, en su caso), EX (ejecución operaciones aritméticas), ME2 (lectura/escritura en memoria) y WB (escritura en el banco de registros). El 10 % de las instrucciones de carga sufren un riesgo estructural en el acceso a la memoria de datos (ME2) con una instrucción aritmética con un operando en memoria (ME1). Salvo por esos riesgos, e ignorando los fallos de antememoria, el CPI alcanzado por el procesador es 1.

La tasa de fallos de las antememorias es de 2 % para la cache de instrucciones y 5 % para la cache de datos. La memoria principal tiene una penalización de 40 ciclos.

Se solicita:

1. Calcular el tiempo de ejecución de un programa P, que ejecuta 2000 millones de instrucciones, sin tener en cuenta los fallos en acceso el acceso a las antememorias.
2. Calcular el número medio de accesos por instrucción (API) a la memoria de instrucciones y a la memoria de datos.
3. Calcular el tiempo de ejecución real del programa P.

Solución:

1. Tiempo de ejecución ideal.
   Hay que tener en cuenta que el 10 % de las instrucciones de carga (15 % del total) sufren un riesgo estructural que obliga a insertar un ciclo de parada:
   CPI = $1 + 0{,}15 \cdot 0{,}1 \cdot 1 = 1{,}015$
   T = 10 ns.
   $T_{ej} = I \cdot CPI \cdot T = (2000 \cdot 10^6) \cdot 1{,}015 \cdot 10^{-9} = 20{,}3$ s.
2. Cálculo de API.
   Todas las instrucciones necesitan acceder a la memoria de instrucciones para buscarse:
   $API_i = 1$ acceso/instrucción
   Las instrucciones de carga (15 %), las de almacenamiento (5 %) y el 25 % de las aritméticas (40 %) necesitan, además, acceder a la memoria de datos:
   $API_d = 0{,}15 + 0{,}05 + (0{,}25 \cdot 0{,}4) = 0{,}3$ accesos/instrucción
   $API = API_i + API_d = 1{,}3$ accesos/instrucción
3. Tiempo de ejecución real.
   $T_{ej} = T_{ej_{ideal}} + T_{memoria}$
   Un programa compuesto por 100 instrucciones realiza por término medio 30 accesos a memoria de datos, con 130 accesos en total. Por lo tanto, la tasa de fallos combinada vendrá dada por:
   $TF_{I+D} = \frac{100}{130} TF_I + \frac{30}{130} TF_D = 0{,}77 \cdot 0{,}02 + 0{,}23 \cdot 0{,}05 = 0{,}0269.$
   $T_{memoria} = I \cdot API \cdot TF \cdot PF \cdot T = (2000 \cdot 10^6) \cdot 1{,}3 \cdot 0{,}0269 \cdot 40 \cdot 10^{-9}$ = 27.98 s
   Sustituyendo:
   $T_{ej} = 20{,}3 + 27{,}98 = 48{,}28$ s
   Otra opción equivalente sería modificar la ecuación del $T_{memoria}$ para que refleje la existencia de antememoria de instrucciones y datos separadas:
   $T_{memoria} = I \cdot ((API_i \cdot TF_i) + (API_d \cdot TF_d)) \cdot PF \cdot T = (2000 \cdot 10^6) \cdot ((1 \cdot 0{,}02) + (0{,}3 \cdot 0{,}05)) \cdot 40 \cdot 10^{-9}$ = 28 s.

□

Ejercicio 3.9.

Un computador tiene los siguientes componentes:

- Un procesador *load/store* semejante al MIPS, con un ancho de palabra de 32 bits y que trabaja a una frecuencia de 100 MHz, con un CPI = 1.3. El 20 % de las instrucciones que ejecuta son lecturas de memoria, y el 10 % son escrituras.
- La memoria cache es de un nivel, separada en 16 KB para instrucciones y 16 KB para datos. El tamaño de bloque es de 16 bytes y la correspondencia es asociativa por conjuntos de dos. Sigue las políticas *write-through* y *no-write allocate*. La tasa de fallos es de 2 % para las lecturas y 5 % para las escrituras.
- Para las escrituras, el procesador dispone de buffers.
- La memoria principal es SDRAM y su temporización en ciclos se describiría como $CL - t_{RCD} - t_{RP} = 1 - 2 - 2$. Se ha medido que en el 90 % de los casos, los accesos encuentran abierta la fila objetivo.

Calcula:

1. La penalización por fallo de lectura, expresada en ciclos de reloj y en segundos. con carga de bloque convencional (esto es, sin aplicar *Critical Word First* ni *Early Restart*–continuar cuanto antes) .
2. La penalización por fallo de lectura, expresada en ciclos de reloj y en segundos, cuando se aplica carga *Critical Word First*.
3. La penalización por fallo de escritura, expresada en ciclos de reloj y en segundos
4. El tiempo de ejecución de un programa de 10 millones de instrucciones, en dos casos:
   - Con carga convencional del bloque
   - Cuando se aplica *Critical Word First*

Solución:

Si los bloques son de 16 bytes y el ancho de palabra de 32 bits, los accesos a la memoria afectarán a cuatro palabras.

1. En el cálculo de la penalización PFL por fallo de lectura, hay que considerar dos casos:
   - Cuando el acceso encuentra abierta la fila objetivo,

$$PFL \text{ (fila abierta)} = L_C + \frac{B}{B_{wc}} = 5 \text{ ciclos} = 50 \text{ ns}$$

   - Cuando el acceso encuentra cerrada la fila objetivo, la penalización será:

$$PFL \text{ (fila cerrada)} = t_{RP} + t_{RCD} + L_C + \frac{B}{B_{wc}} = 9 \text{ ciclos} = 90 \text{ ns}$$

   - El promedio será de

$$PFL \text{ (promedio)} = 0{,}9 \times 5 + 0{,}1 \times 9 = 5{,}4 \text{ ciclos} = 54 \text{ ns}$$

2. La penalización por fallo de lectura con *Critical Word First* es idéntica a la de un bloque de sólo una palabra. Por tanto:
   - Cuando el acceso encuentra abierta la fila objetivo, la penalización será

$$PFL \text{ (fila abierta)} = L_C + \frac{B}{B_{wc}} = 2 \text{ ciclos} = 20 \text{ ns}$$

- Cuando el acceso encuentra cerrada la fila objetivo, la penalización será:

$$PFL \text{ (fila cerrada)} = t_{RP} + t_{RCD} + L_C + \frac{B}{B_{wc}} = 6 \text{ ciclos} = 60 \text{ ns}$$

- El promedio será de

$$PFL \text{ (promedio)} = 0{,}9 \times 2 + 0{,}1 \times 6 = 2{,}4 \text{ ciclos} = 24 \text{ ns}$$

3. No hay ninguna penalización por fallo de escritura, pues con la política *no-write allocate* y los buffers de escritura, el procesador nunca ha de esperar para transferencias con memoria.
4. El tiempo de ejecución del programa de 10 millones de instrucciones será

$$t = I \times CPI \times T + I \times LPI \times TFL \times PFL$$

- Con carga convencional del bloque:

$$\begin{aligned} T &= 10 \cdot 10^6 \text{ instrucciones} \times 1{,}3 \text{ ciclos/instrucción} \times 10 \cdot 10^{-9} \text{ s/ciclo} + \\ &+ 10 \cdot 10^6 \text{ instrucciones} \times 1{,}2 \text{ lecturas/instrucción} \times \\ &\quad \times 0{,}02 \text{ fallos/lectura} \times 54 \cdot 10^{-9} \text{ s/fallo} \\ &\simeq 130 + 13 = 143 \text{ ms} \end{aligned}$$

- Cuando se aplica *Critical Word First* sólo cambia la penalización por fallo, y resulta:

$$T = 130 + 10 \cdot 10^6 \times 1{,}2 \times 0{,}02 \times 24 \simeq 130 + 6 = 136 \text{ ms}$$

□