COGNOMS:

NOM: DNI/NIE:

IMPORTANTE leer atentamente antes de empezar el examen: Escriba los apellidos, el nombre y el DNI/NIE antes de empezar el examen. Escriba un solo carácter por recuadro, en mayúsculas y lo más claramente posible. Es importante que no haya tachones ni borrones y que cada carácter quede enmarcado dentro de su recuadro sin llegar a tocar los bordes. Use un único cuadro en blanco para separar los apellidos y nombres compuestos si es el caso. No escriba fuera de los recuadros, todo lo que haya fuera de ellos es ignorado. La identificación del alumno se realiza de forma automática, no seguir correctamente estas instrucciones puede comportar no tener nota.

Problema 1. (3 puntos)

Se dispone de un computador que tiene direcciones lógicas de 64 bits y direcciones físicas de 34 bits. La jerarquía de memoria está compuesta de una memoria principal y dos niveles de memoria cache con las siguientes características.

- Una memoria principal de 16 GB distribuidos en 4 módulos DIM. Cada módulo DIM está compuesto de 8 chips. Cada chip contiene 8 bancos de 64K filas y 1K columnas de 1 byte.
- La cache L2 tiene un tamaño de 512 KBytes y es 8-asociativa. La política de escritura es Copy Back-Write Allocate. El tamaño de bloque es de 256 bytes. Tasa de fallos=0,1. Tsa= 6 ciclos. La lectura/escritura de un bloque de MP tarda 130 ciclos (incluyendo el envío del dato a la CPU en caso de lectura).
- La cache L1I de instrucciones tiene un tamaño de 8 Kbytes y es 2-asociativa. El tamaño de línea es de 128 bytes. Tasa de fallos=0,2. Tsa= 1ciclo
- La cache L1D de datos tiene un tamaño de 8 Kbytes y es 4-asociativa. La política de escritura es Write through-Write NO Allocate. El tamaño de línea es de 128 bytes. Tasa de fallos=0,3. Tsa= 1ciclo
- El sistema dispone además de dos TLB de 16 entradas completamente asociativos, uno para cada cache L1, con Tsa=1 ciclo, que se acceden en paralelo junto con la L1 correspondiente.
- El tamaño de página del sistema es de 4 KBytes.

a) **Calcula** el número de líneas, vías y conjuntos que tiene la cache L1D. **Especifica claramente** cómo has realizado los cálculos.

b) **Calcula** el tamaño del TLB de datos en bits. **Especifica claramente** cómo has realizado los cálculos.

La CPU lanza un acceso a la dirección lógica 0xEFABCD0123456789, que el TLB traduce como 0x3BCDE789. El dato al que quiere acceder la CPU no esta en la L1D, y se encuentra en la memoria cache L2.

c) **Indica** en qué conjunto de la L2 se encuentra el dato y cuál es la etiqueta que se guardará en memoria cache L2. **Justifica** la respuesta.

Se ejecuta la instrucción movl (%eax), %ebx, cuya ejecución en el procesador (sin contar el tiempo de acceso a la memoria de datos ni a la memoria de instrucciones) tarda diez ciclos. La ejecución de la instrucción provoca los siguientes eventos.

- Acierto de TLB de instrucciones para acceder a la instrucción
- Acierto en la cache L1 de instrucciones
- Acierto de TLB de datos para acceder al dato
- Fallo en la cache L1 de datos
- Fallo en L2 (el bloque reemplazado tiene el dirty bit a 0)
- Acceso a MP, donde se encuentra el dato buscado

d) **Calcula** el tiempo total de ejecución de la instrucción movl (%eax), %ebx. **Especifica claramente** cómo has calculado el tiempo

e) **Indica** el tamaño máximo que puede tener la cache L1D para que sea posible acceder a la cache L1D y al TLB asociado en paralelo, suponiendo que se mantiene el tamaño de línea y el grado de asociatividad, y que se mantienen también el resto de parámetros de la jerarquía de memoria. **Justifica** la respuesta.

COGNOMS:

NOM: DNI/NIE:

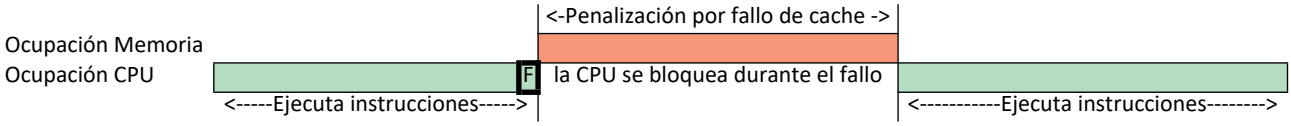
Problema 2. (3.5 puntos)

Se ha simulado la ejecución de un programa P en un procesador que denominaremos IDEAL. En este procesador IDEAL no hay ninguna penalización por fallo de cache. De esta simulación se ha obtenido que el programa P se ha ejecutado en $5x10^9$ ciclos durante los que ha ejecutado $2x10^9$ instrucciones, de las que $500x10^6$ son instrucciones de acceso a datos (Load/Store) y se han producido $50x10^6$ fallos en la cache de datos (los fallos en la cache de instrucciones son negligibles, con lo que los ignoraremos durante todo el problema). Suponemos que la probabilidad de fallar en cualquier ciclo es la misma y es independiente de que se haya fallado o no en el ciclo anterior.

a) **Calculad** el CPI de P en el procesador IDEAL (CPI_{IDEAL}).

b) **Calculad** el número medio de ciclos transcurridos entre 2 fallos.

El mismo programa lo ejecutamos en un procesador real con las mismas características que el IDEAL con la única diferencia que en caso de fallo en la cache de datos se bloquea la ejecución de instrucciones durante un cierto numero de ciclos que corresponden al tiempo medio de penalización por fallo de cache (Tpf) hasta que se resuelve el fallo. La siguiente figura ilustra este hecho cuando se detecta un fallo (F).

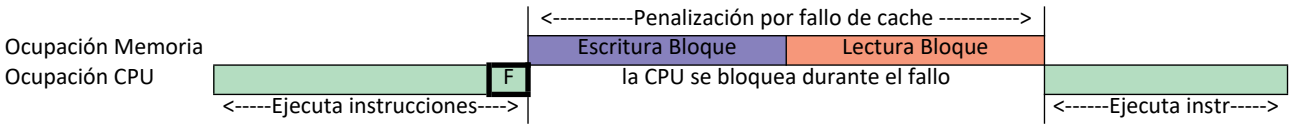


El programa P se ha ejecutado en el procesador real (que llamaremos procesador A) en 3 segundos. Este procesador A funciona a una frecuencia de 2 GHz.

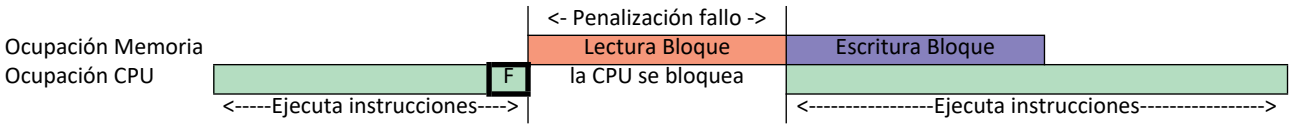
c) **Calculad** el CPI de P en el procesador A (CPI_A)

d) **Calculad** el número medio de ciclos de penalización por fallo de cache (Tpf)

La cache de datos sigue una política de escritura COPY BACK + WRITE ALLOCATE. En caso de fallo, si la línea reemplazada ha sido modificada, la penalización es de 30 ciclos, mientras que si no lo ha sido es sólo de 15 ciclos. Se sabe que durante la ejecución de P, en media 1/3 de los bloques reemplazados han sido modificados (dirty bit = 1). La siguiente figura muestra el cronograma de un fallo en que la línea reemplazada ha sido modificada.



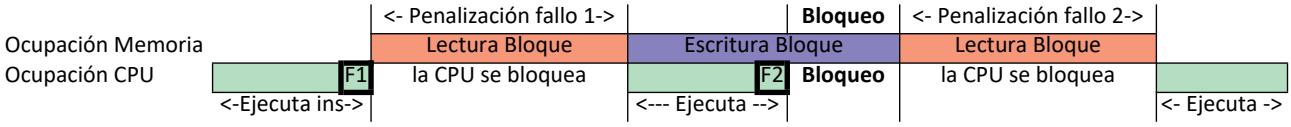
Una posible mejora (vista en clase de teoría) consiste en incorporar un buffer para almacenar el bloque reemplazado, de forma que podamos leer primero el bloque que ha provocado el fallo y a continuación escribir en memoria el bloque reemplazado. Esta implementación permite que el procesador pueda seguir ejecutando instrucciones y la cache pueda ser accedida durante la escritura del bloque reemplazado a memoria, tal como muestra la siguiente figura.



De momento supondremos la situación ideal (aunque no realista) en que nunca se produce un fallo de cache mientras se realiza la escritura del bloque reemplazado. A este procesador lo denominaremos procesador B

e) **Calculad** el numero de ciclos necesario para ejecutar P en el procesador B

En la implementación real (que denominamos procesador C) dispondremos de un buffer de una sola entrada que nos permite tener como máximo una escritura pendiente. Si durante la escritura del bloque causada por un fallo (F1) se produce un segundo fallo (F2), hay que esperar a que la jerarquía de memoria complete la escritura del bloque antes de que pueda empezar a servir el siguiente fallo, con lo que el procesador se bloqueará por unos ciclos adicionales (**Bloqueo**), tal como muestra la siguiente figura (en el ejemplo suponemos que el siguiente fallo reemplaza un bloque no modificado).



Durante la fase en que la CPU ejecuta instrucciones estas se ejecutan con la misma distribución que en el procesador IDEAL, por lo que el número medio de ciclos en que la CPU ejecuta instrucciones (sin contar el bloqueo debido a la lectura del bloque solicitado por el fallo F1) entre F1 y F2 será el mismo.

f) **Calculad** la probabilidad de que se produzca un segundo fallo durante la escritura de un bloque reemplazado

Hemos medido que, si se produce un segundo fallo durante el intervalo de escritura de un bloque anterior, en media la CPU se bloquea durante 7 ciclos (ciclos correspondientes al intervalo **Bloqueo** de la figura anterior) antes de que se pueda iniciar la lectura del bloque que ha causado el segundo fallo.

g) **Calculad** el numero de ciclos necesario para ejecutar P en el procesador C

COGNOMS:

NOM: DNI/NIE:

Problema 3. (3.5 puntos)

Tenemos un procesador con direcciones físicas de 20 bits y unidad mínima de direccionamiento un byte. Dicho procesador dispone de una cache L1 de 8 Kbytes, 2-asociativa, con algoritmo de reemplazo LRU y tamaño de bloque (linea) de 32 bytes.

En los siguientes apartados se asume que todos los accesos son **lecturas** a **byte**. En todas las tablas se describen los campos necesarios. Si un campo no se describe es porque ya se ha descrito en tablas anteriores. Para representar un valor en hexadecimal con un numero de bits no múltiplo de 4, asume ceros en los de más peso (101000 -> 28).

a) **Dibuja** una dirección física con los campos de bits relevantes para el acceso a la cache L1, indicando claramente el nombre y tamaño de cada uno de ellos.

b) **Rellena** la siguiente tabla a partir de la secuencia de referencias a memoria dada:

@física: Dirección física (en hexadecimal)
BM: Bloque de memoria que se accede (en hexadecimal).
byte: Byte dentro del bloque (en hexadecimal).
TAG: Etiqueta del acceso a cache (en hexadecimal).
CJT: Conjunto de cache en que se mapea el bloque (en hexadecimal).
A/F: Acierto de Cache L1 (A) o Fallo de Cache L1 (F).
LMP: Numero de bytes leídos de memoria principal (**vacío si no se lee de MP**) (en decimal).
Inicialmente la cache está vacía.

@física	BM	byte	TAG	CJT	A/F	LMP
22000						
44001						
22002						
88003						
44004						
44025						
44046						
44067						

(sigue atrás)

Esta cache implementa *prefetch hardware* mediante un *buffer de prefetch* de un bloque de capacidad. El funcionamiento del *buffer de prefetch* es el siguiente:

- En caso de acierto en cache no se hace *prefetch*.
- En caso de fallo de cache al bloque (**i**):
  - Si el bloque (**i**) esta en el *buffer* se mueve a la cache y se realiza *prefetch* del bloque (**i+1**) al buffer.
  - Si el bloque (**i**) no está en el *buffer* se accede a memoria principal para buscar el bloque (**i**) y además se realiza *prefetch* del bloque (**i+1**) al *buffer.*

c) **Rellena** la siguiente tabla a partir de la secuencia de referencias a memoria dada:

BEB: Bloque de memoria presente en el buffer (en hexadecimal) (**vacío si no hay nada**)
BPMP: Bloque de memoria que se prebusca (en hexadecimal) (**vacío si no hay prefetch**).
A/F B: Acierto (A) o Fallo (F) en el buffer de prefetch (**vacío si no hay prefetch**).
LMP: Numero de bytes leídos de memoria principal (**vacío si no se lee de MP**) (en decimal).
Inicialmente la cache y el buffer están vacíos. La secuencia de referencias es la misma que en la tabla anterior.

@física	BM	byte	TAG	CJT	A/F	BEB	BPMP	A/F B	LMP
22000									
44001									
22002									
88003									
44004									
44025									
44046									
44067									

En este procesador todos los aciertos a cache tardan un ciclo, mientras que los fallos introducen una penalización media de P ciclos. Se ha ejecutado una aplicación A en $5x10^9$ ciclos para lo que se han realizado $1x10^9$ accesos a memoria. Para la aplicación A la cache tienen una tasa de fallos del 5%.

Se desea añadir un predictor de vía a la cache. El predictor de vía seleccionado tiene una tasa de aciertos del 90% para la aplicación A. La cache con predictor de vía añade un ciclo de penalización en caso de fallo de predictor, tanto si es acierto como si es fallo de cache, pero permite aumentar la frecuencia del procesador en un 10%.

d) **Calcula** los ciclos que tardaría la aplicación A con el predictor de vía.

e) **Calcula** la ganancia en tiempo de ejecución (speed-up) del procesador con predictor de vía (respecto al que no implementa predictor de vía) para la aplicación A. Da el resultado en %.