PRÁCTICA 4

Javier Martín Plaza 77155563Q

Jun Jie Li Zhu (David) – 26546449X

Ejercicio 1.

En este ejercicio realizaremos un estudio de la tasa de fallos según el tamaño de bloque y el tamaño de caché, a continuación, se presentarán unos diagramas, a partir del estudio realizado y comentaremos las características documentando la evolución para cada uno de los casos.

Correspondencia Directa

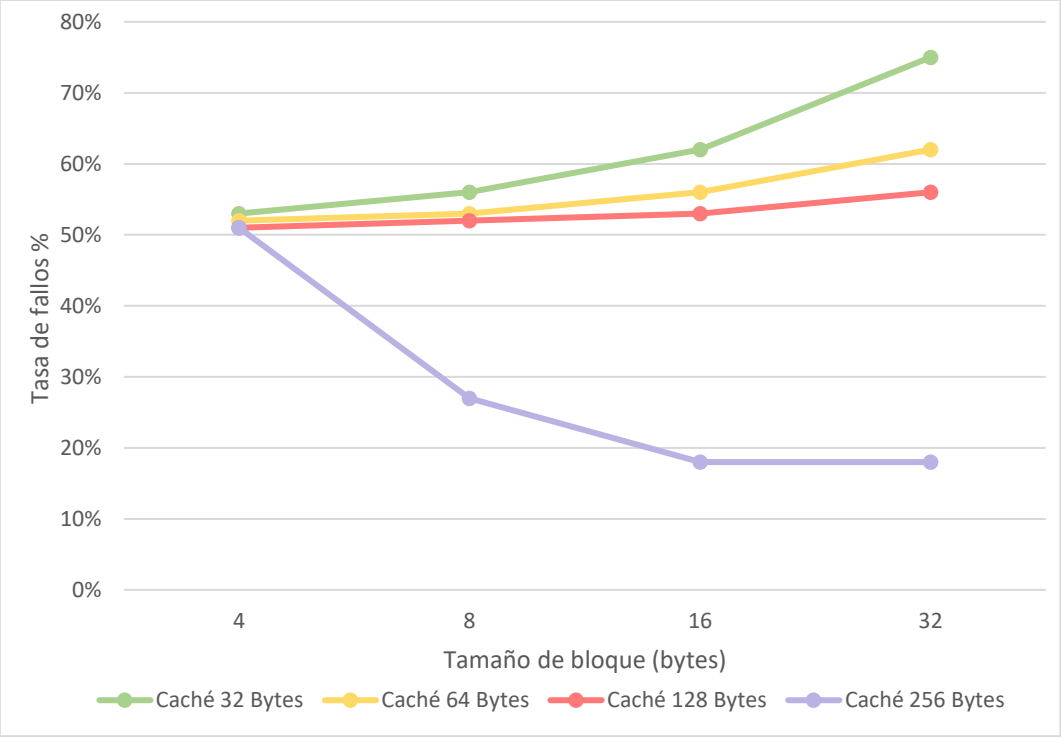


Ilustración 1: Tasa de Fallos (%), respecto al tamaño de bloque (bytes). Correspondencia Directa.

Respecto a la Correspondencia directa podemos observar que tanto en las cachés de 32, 64 y 128 bytes, tienen un crecimiento progresivo en porcentaje de fallos respecto al tamaño de bloque que hay, sin embargo, en la caché de 256 bytes la tasa de fallo empieza a decrementar conforme al tamaño. En términos generales podemos entender que a cuanto mas tamaños de bloque reducimos la tasa de fallo como bien vemos en la caché de 256, el problema llega que hay diversos problemas y no siempre se cumple esa regla y es que como ocurre en el resto de casos llega un punto en el que número de bloques que puede albergar la caché disminuye, desciendo la localidad espacial y aumentando la tasa de fallo.

Tabla de tamaño de bloque por tamaño de caché, con su respectiva tasa de fallos, en cada caso:

Tamaño de bloque	Caché 32 Bytes	Caché 64 Bytes	Caché 128 Bytes	Caché 256 Bytes
4	53%	52%	51%	51%
8	56%	53%	52%	27%
16	62%	56%	53%	18%
32	75%	62%	56%	18%

Datos recopilados:

Para Caché 32 Bytes

Número de bloques	Tamaño de bloque	Tasa de fallos
8	1	53%
4	2	56%
2	4	62%
1	8	75%

Para Caché 64 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
16	1	52%
8	2	53%
4	4	56%
2	8	62%

Para Caché 128 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
32	1	51%
16	2	52%
8	4	53%
4	8	56%

Para Caché 256 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
64	1	51%
32	2	27%
16	4	18%
8	8	18%

Asociativa por conjuntos de 2 vías (LRU)

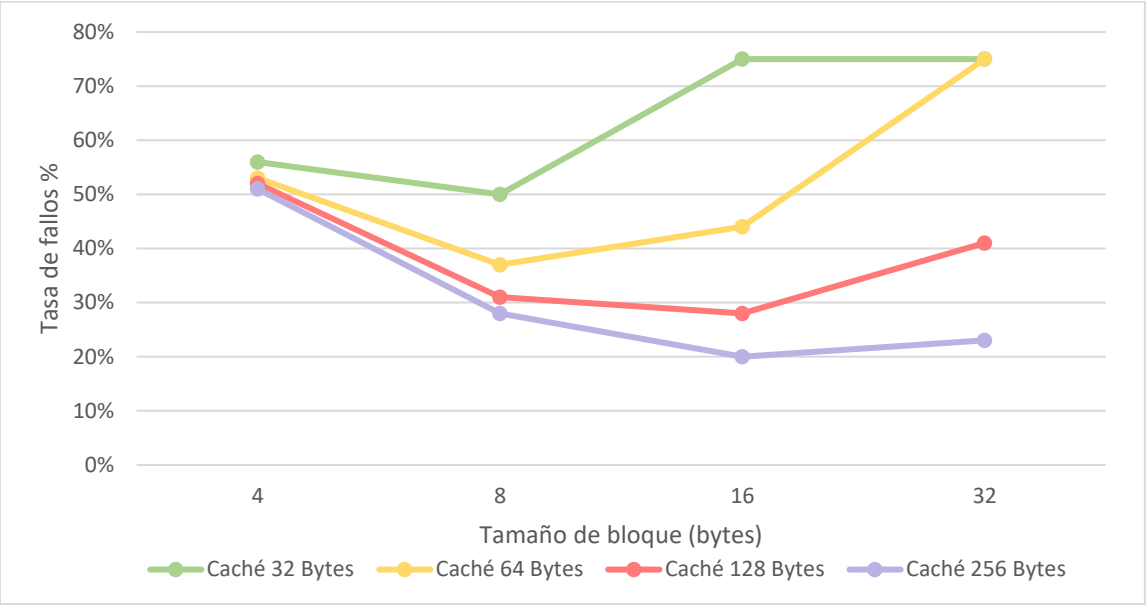


Ilustración 2: Tasa de fallos (%), respecto al tamaño de bloque (bytes). Asociativa por conjuntos de 2 vías (LRU).

Respecto al diagrama anterior podemos observar un caso similar al anterior y es que en cada cache empieza funcionando la relación de disminución de tasa de fallos respecto al tamaño de bloque, debido a que permite en la asociativa por conjuntos de 2 vías coger bloques con posiciones contiguas permitiendo un acceso más rápido al dato sin caer en fallo, hasta que llega un punto donde la cache no puede contener más números de bloques aumentando considerablemente de nuevo el porcentaje respecto a la tasa de fallos. Debemos de tener en cuenta de que el número coloreado en rojo en la caché de 32 bytes es debido a que no se puede 2 vías de 1 numero de bloque con un tamaño de 8, esto ocurrirá de nuevo en el próximo apartado.

Tabla de tamaño de bloque por tamaño de caché, con su respectiva tasa de fallos, en cada caso:

Tamaño de bloque	Caché 32 Bytes	Caché 64 Bytes	Caché 128 Bytes	Caché 256 Bytes
4	56%	53%	52%	51%
8	50%	37%	31%	28%
16	75%	44%	28%	20%
32	75%	75%	41%	23%

Datos recopilados:

Para Caché 32 Bytes

Número de bloques	Tamaño de bloque	Tasa de fallos
8	1	56%
4	2	50%
2	4	75%
1	8	75%

Para Caché 64 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
16	1	53%
8	2	37%
4	4	44%
2	8	75%

Para Caché 128 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
32	1	52%
16	2	31%
8	4	28%
4	8	41%

Para Caché 256 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
64	1	51%
32	2	28%
16	4	20%
8	8	23%

Asociativa por conjuntos de 4 vías (LRU)

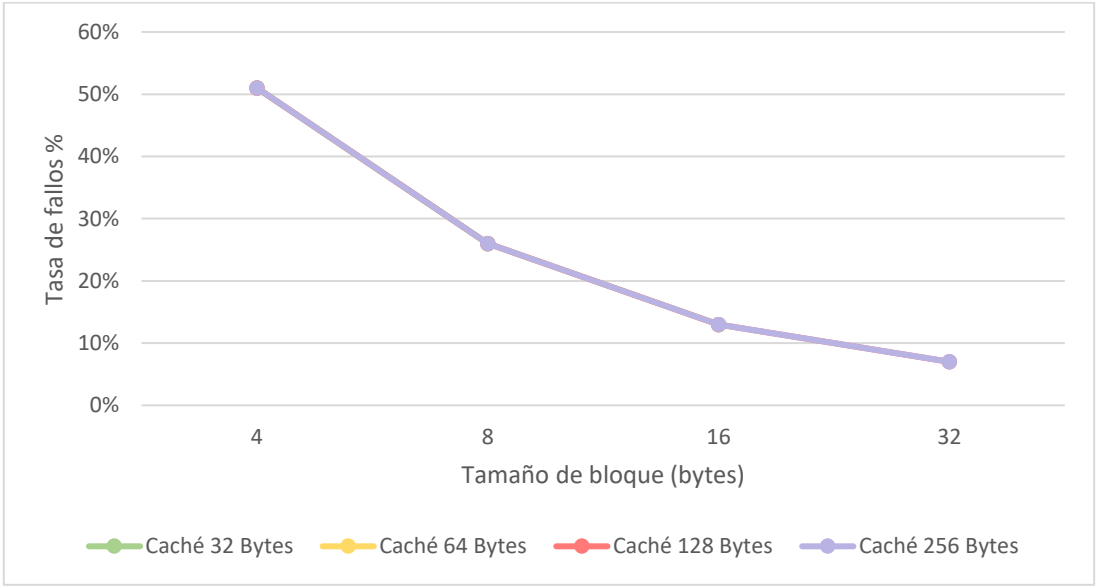


Ilustración 3: Tasa de fallos (%), respecto al tamaño de bloque (bytes). Asociativa por conjuntos de 4 vías (LRU).

En este caso podemos observar perfectamente con las cachés con más tamaño son capaces de albergar el numero de bloques permitiendo llevarse consigo bloque de posiciones cercanas y aumentando el numero de bloques nos permite reducir la tasa de fallo. Como en el caso anterior nos ocurre que no podemos realizar tres tasas de fallos en las cachés con menos bytes debido a que no tenemos vías suficientes conforme al número de bloques y a su tamaño donde poder sacar un porcentaje respecto a la tasa de fallo. As que omitiremos reproducir estos casos en la gráfica.

Tabla de tamaño de bloque por tamaño de caché, con su respectiva tasa de fallos, en cada caso:

Tamaño de bloque	Caché 32 Bytes	Caché 64 Bytes	Caché 128 Bytes	Caché 256 Bytes
4	51%	51%	51%	51%
8	26%	26%	26%	26%
16	75%	13%	13%	13%
32	75%	75%	7%	7%

Datos recopilados:

Para Caché 32 Bytes

Número de bloques	Tamaño de bloque	Tasa de fallos
8	1	51%
4	2	26%
2	4	75%
1	8	75%

Para Caché 64 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
16	1	51%
8	2	26%
4	4	13%
2	8	75%

Para Caché 128 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
32	1	51%
16	2	26%
8	4	13%
4	8	7%

Para Caché 256 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
64	1	51%
32	2	26%
16	4	13%
8	8	7%

Totalmente Asociativa

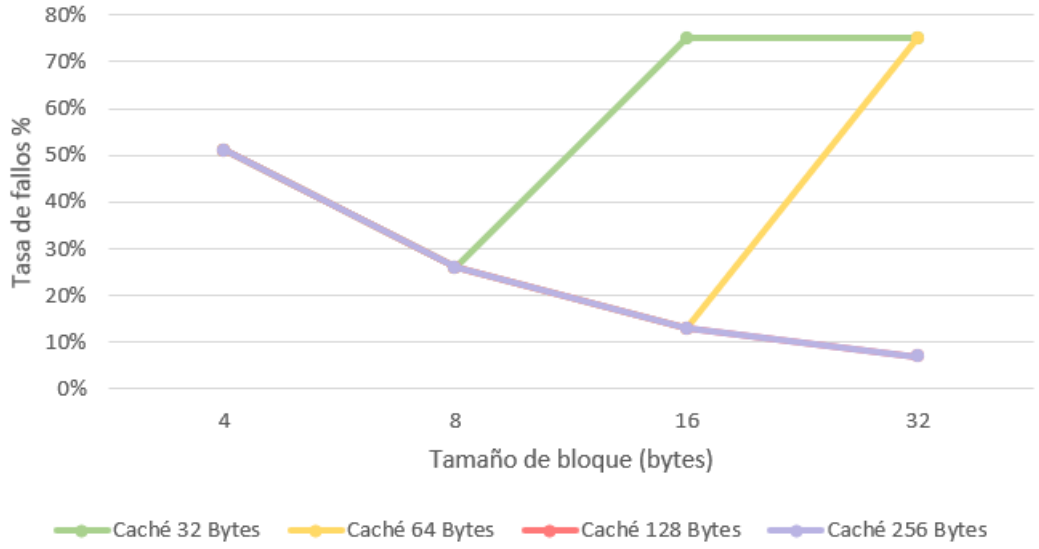


Ilustración 4: Tasa de fallos (%), respecto al tamaño de bloque (bytes). Totalmente Asociativa (LRU).

Como vemos en la totalmente asociativa tenemos un caso totalmente similar a la asociativa por conjuntos de n vías, en este caso las cachés de mayor tamaño son capaces de contener los números de bloques y a cuantos más, mas tasa de acierto obtendremos, y en el caso de las cachés de tamaño mas pequeño, al principio tienen un decrecimiento de fallos hasta que no pueden tener más números de bloques, y conforme aumentas los números de bloques termina al final aumentando la tasa de fallos. Cabe destacar que la política totalmente asociativa no tiene fallos de conflictos debido a que cada bloque tiene asociado su propio dato evitando así que intercepten datos entre mismos bloques.

Tabla de tamaño de bloque por tamaño de caché, con su respectiva tasa de fallos, en cada caso:

BLOCK SIZE	Caché 32 Bytes	Caché 64 Bytes	Caché 128 Bytes	Caché 256 Bytes
4	51%	51%	51%	51%
8	26%	26%	26%	26%
16	75%	13%	13%	13%
32	75%	75%	7%	7%

Datos recopilados:

Para Caché 32 Bytes

Número de bloques	Tamaño de bloque	Tasa de fallos
8	1	51%
4	2	26%
2	4	75%
1	8	75%

Para Caché 64 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
16	1	51%
8	2	26%
4	4	13%
2	8	75%

Para Caché 128 Bytes:

Número de bloques	Tamaño de bloque	Tasa de fallos
32	1	51%
16	2	26%
8	4	13%
4	8	7%

Para Caché 256 Bytes:

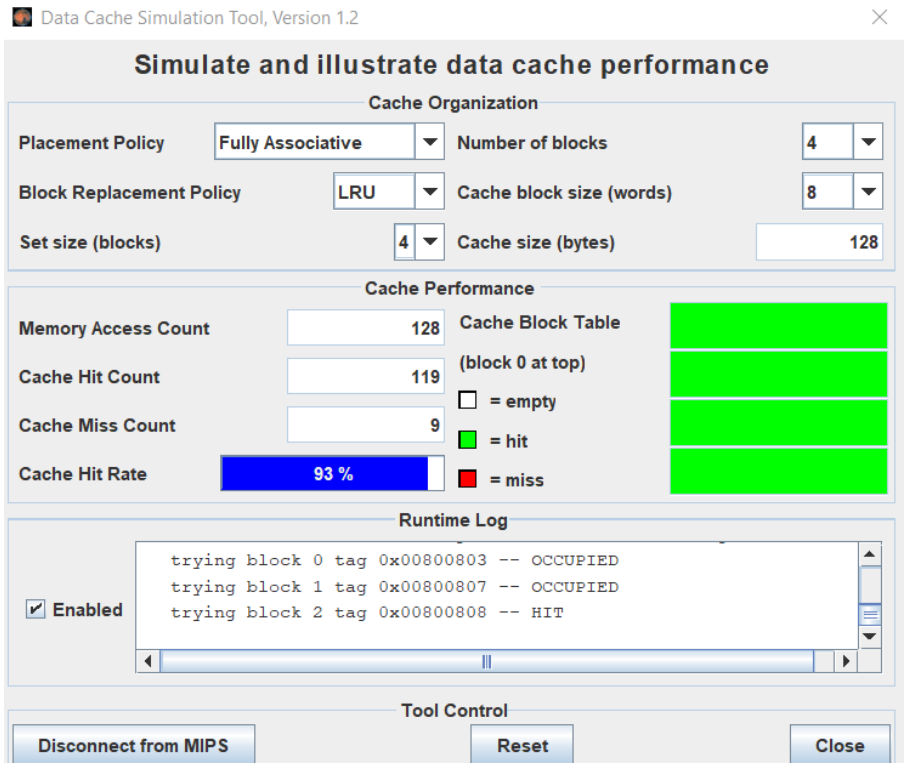
Número de bloques	Tamaño de bloque	Tasa de fallos
64	1	51%
32	2	26%
16	4	13%
8	8	7%

Ejercicio 2.

Al ir configurando el simulador de datos de caché hemos podido analizar la evolución que ha tenido cada política permitiéndonos observar que configuración nos permitía obtener una mayor eficiencia beneficiados por un mayor ratio de acciertos, en primer lugar se nos pedía una caché de 128 bytes y que tuviera la mayor tasa de aciertos, para ello observamos el código y nos encontramos con tres variables, las primeras dos, A y B estaban compuestas por cuatro arrays, de los cuales cada uno tenia 8 palabra, datos, en este caso valores int. Si hacemos una multiplicación del tamaño por la cantidad de valores, obtendremos

Según el ejercicio de producto escalar tenemos dos variables que contienen 4 arrays, en los que cada uno tiene 8 datos insertados, debemos aplicar una configuración de caché concorde con el ejercicio, con un tamaño de 128 Bytes; para que tengamos una buena eficiencia y tengamos un mayor ratio de aciertos a la hora de utilizar nuestra caché. Para ello

multiplicaremos la cantidad de datos de cada array por 4 bytes el cual es su tamaño, este nos da 32 bytes por array si lo hacemos por los cuatro arrays obtendremos 128 bytes por cada variable. Con esto primero deducimos que son bloques multipalabra para ello nuestra configuración idónea debería ser que llevase consigo bloques a cache que estuvieran a continuación del bloque solicitado para obtener un mejor acceso a estos datos evitando fallos de compulsory. Estudiando cada caso nos hemos percatado de que la configuración más eficiente es la de la política de totalmente asociado, entre otras cosas porque evitamos los errores de conflictos, a parte usaremos la siguiente configuración en concreto cuatro números de bloques multipalabras de 8 de cantidad. Esto nos permite obtener un 93% en tasa de aciertos, gracias al principio de localidad espacial, debido a la gran cantidad de datos que vamos a manejar. No obstante, usaremos la política de LRU, least recently used, bloque menos usado recientemente que obtiene una mayor tasa de aciertos ante el ramdom que obtiene un 1% menos de aciertos debido a posible fallo de acceso. Por lo tanto, podemos deducir que a mayor tamaño de bloque en términos generales siguiendo la graficas anteriores existe una mayor tasa de acierto.



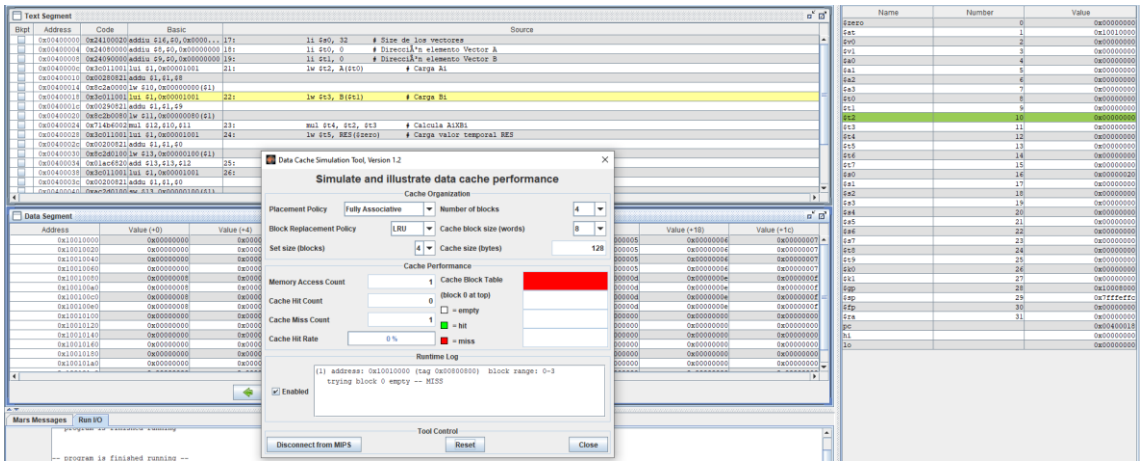
Ejercicio 3.

De un total de 128 accesos obtenemos 9 fallos.

- ➔ Los 4 primeros fallos son de tipo obligatorio (coldstart), ya que la caché está vacía y es la primera vez que accedo a ese bloque de datos.
- ➔ Los 5 siguientes fallos podemos observar que los bloques están ocupados. Mi caché no es lo suficientemente grande como para almacenar más bloques y por lo tanto tenemos que remplazar esos bloques de ahí.
- ➔ No hay ningún fallo de tipo conflicto puesto que hacemos uso de la política totalmente asociativa no tiene fallo de conflicto.

A continuación mostraremos unas capturas de cada uno de los fallos aparecidos.

Fallo 1

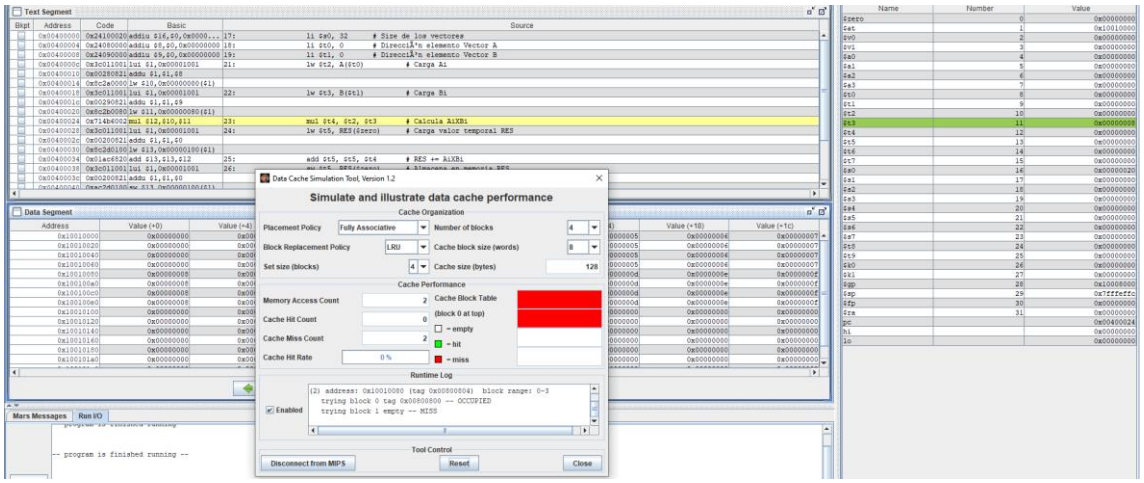


(1) address: 0x10010000 (tag 0x00800800) block range: 0-3

trying block 0 empty -- MISS

Fallo por Compulsory.

Fallo 2



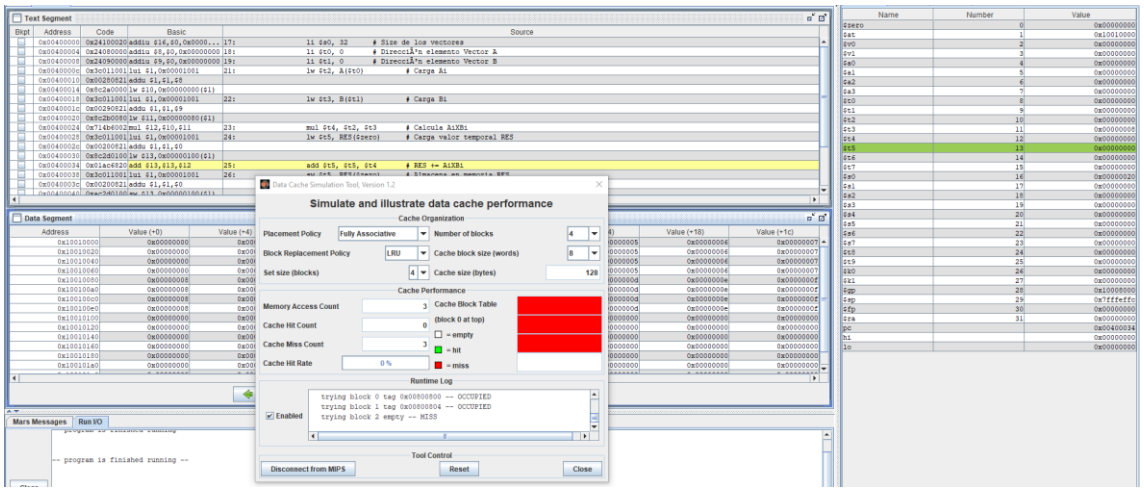
(2) address: 0x10010080 (tag 0x00800804) block range: 0-3

trying block 0 tag 0x00800800 -- OCCUPIED

trying block 1 empty – MISS

Fallo por Compulsory.

Fallo 3



(3) address: 0x10010100 (tag 0x00800808) block range: 0-3

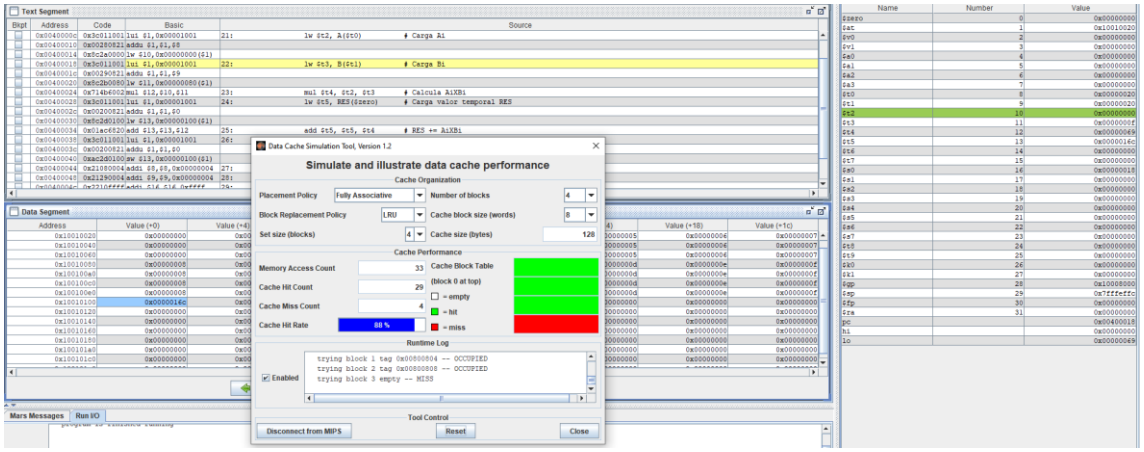
trying block 0 tag 0x00800800 -- OCCUPIED

trying block 1 tag 0x00800804 -- OCCUPIED

trying block 2 empty -- MISS

Fallo por Compulsory.

Fallo 4



(33) address: 0x10010020 (tag 0x00800801) block range: 0-3

trying block 0 tag 0x00800800 -- OCCUPIED

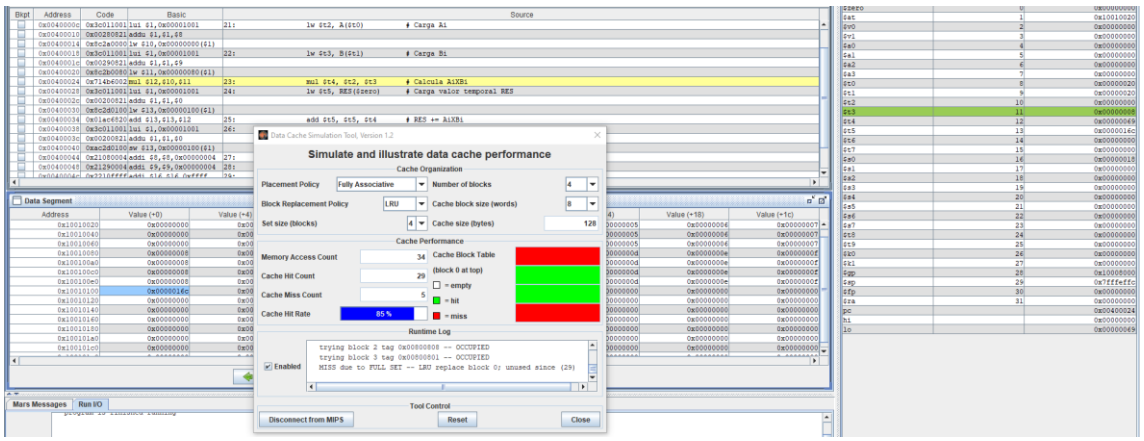
trying block 1 tag 0x00800804 -- OCCUPIED

trying block 2 tag 0x00800808 -- OCCUPIED

trying block 3 empty – MISS

Fallo por Compulsory.

Fallo 5



(34) address: 0x100100a0 (tag 0x00800805) block range: 0-3

trying block 0 tag 0x00800800 -- OCCUPIED

trying block 1 tag 0x00800804 -- OCCUPIED

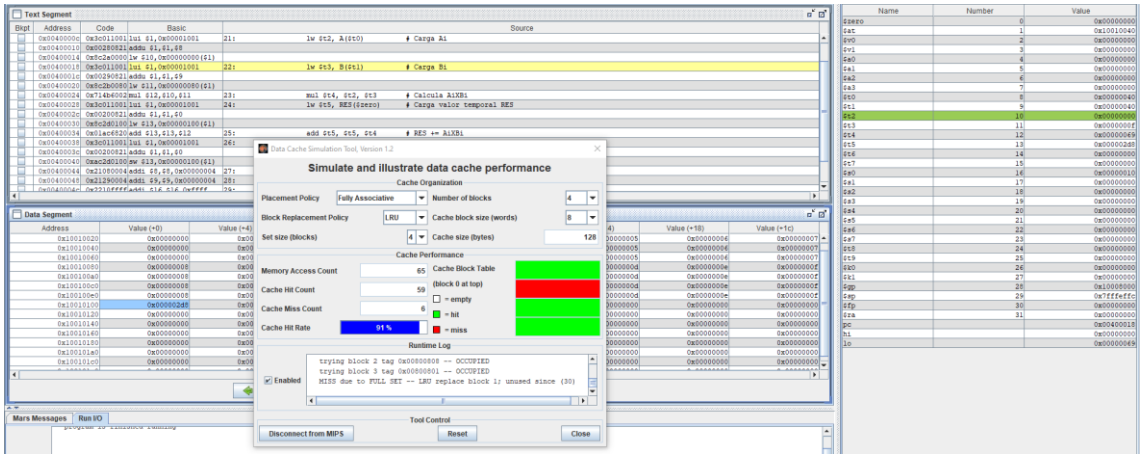
trying block 2 tag 0x00800808 -- OCCUPIED

trying block 3 tag 0x00800801 -- OCCUPIED

MISS due to FULL SET -- LRU replace block 0; unused since (29)

Fallo por Capacity.

Fallo 6



(65) address: 0x10010040 (tag 0x00800802) block range: 0-3

trying block 0 tag 0x00800805 -- OCCUPIED

trying block 1 tag 0x00800804 -- OCCUPIED

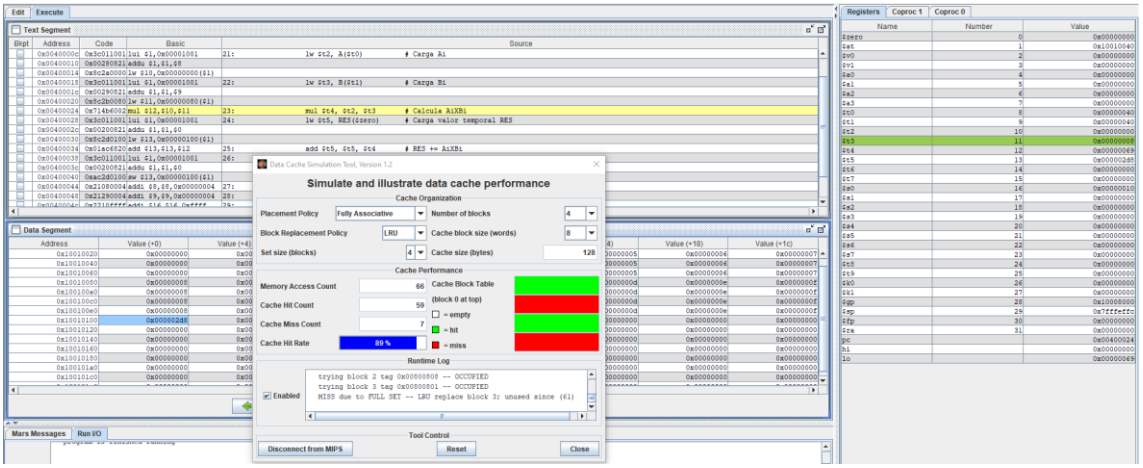
trying block 2 tag 0x00800808 -- OCCUPIED

trying block 3 tag 0x00800801 -- OCCUPIED

MISS due to FULL SET -- LRU replace block 1; unused since (30)

Fallo por Capacity.

Fallo 7



(66) address: 0x100100c0 (tag 0x00800806) block range: 0-3

trying block 0 tag 0x00800805 -- OCCUPIED

trying block 1 tag 0x00800802 -- OCCUPIED

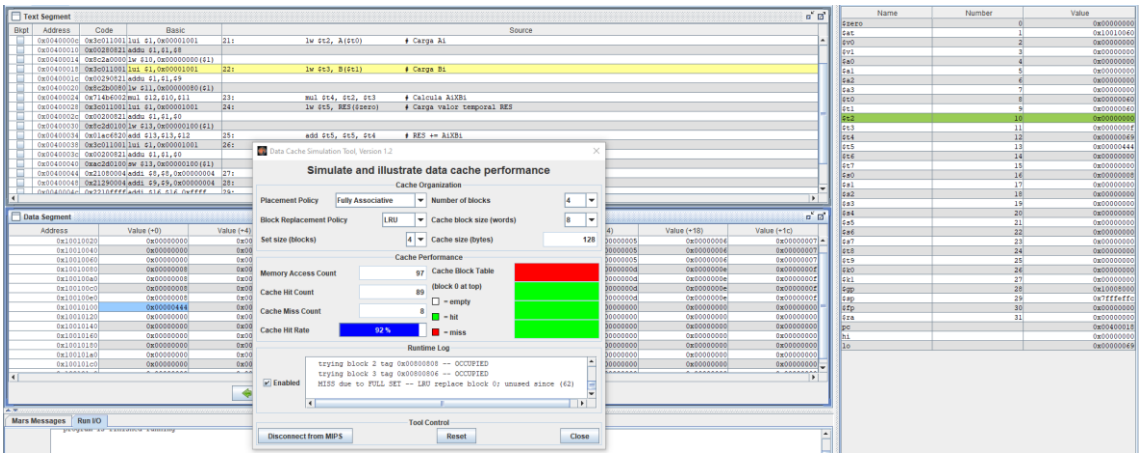
trying block 2 tag 0x00800808 -- OCCUPIED

trying block 3 tag 0x00800801 -- OCCUPIED

MISS due to FULL SET -- LRU replace block 3; unused since (61)

Fallo por Capacity.

Fallo 8



(97) address: 0x10010060 (tag 0x00800803) block range: 0-3

trying block 0 tag 0x00800805 -- OCCUPIED

trying block 1 tag 0x00800802 -- OCCUPIED

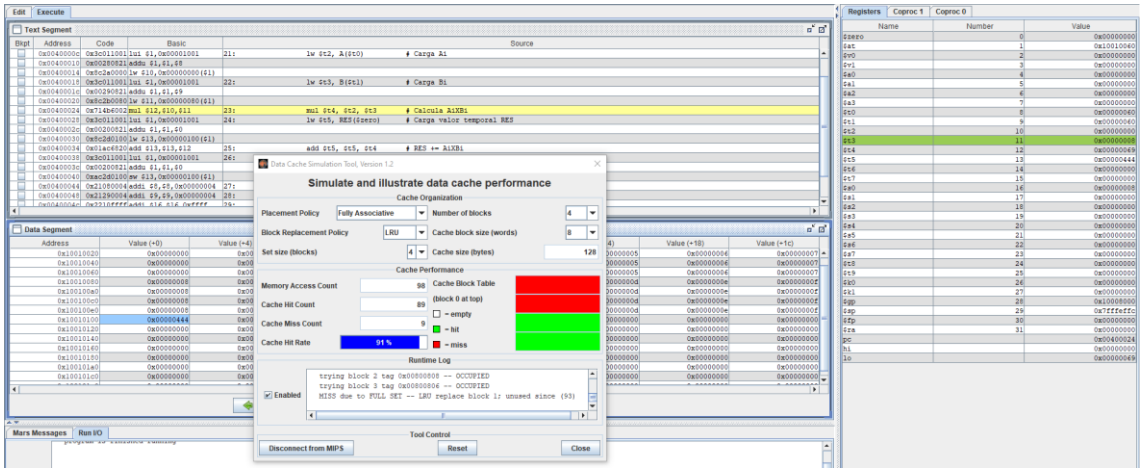
trying block 2 tag 0x00800808 -- OCCUPIED

trying block 3 tag 0x00800806 -- OCCUPIED

MISS due to FULL SET -- LRU replace block 0; unused since (62)

Fallo por Capacity.

Fallo 9



(98) address: 0x100100e0 (tag 0x00800807) block range: 0-3

trying block 0 tag 0x00800803 -- OCCUPIED

trying block 1 tag 0x00800802 -- OCCUPIED

trying block 2 tag 0x00800808 -- OCCUPIED

trying block 3 tag 0x00800806 – OCCUPIED

MISS due to FULL SET -- LRU replace block 1; unused since (93)

Fallo por Capacity.