**Note:**  (Late: -15% Penalty).

1. (8 pts) Suppose a computer using direct mapped cache has 232 bytes of byte-addressable main memory, and a cache of 32 blocks, where each cache block contains 16 bytes.
   1. How many blocks of main memory are there?

232 / 24 = 228

* 1. What are the **sizes** of the tag, block, and offset fields (the form of a memory address as seen by the cache) ?

( Please answer your sizes in powers of 2…)

Answer: 32-bit address, given 32 (25) blocks, 16 (24) bytes per block

Tag: 232 – 25 - 24 = 223

Block: 25

Offset: 24

* 1. To which cache block will the memory address 0x0000DB63 map?

Answer:

Byte Address = Tag Block Offset

0x0000DB63 = 0000000000001100101 10110 0011

Block = 10110 which implies Block 22 (or block 0x16h)

1. (8 pts) Suppose you are architecting a 32-bit system. The chip designers have a limited your cache size to 4MB of SRAM. You have chosen to use a 4-way set associative cache with 32 bytes per cache block.

What is the format of a memory address as seen by the cache?

(Hint: Because this architecture is byte-addressable, and the number of addresses is critical in determining the address format, you must convert everything to bytes.)

Show your work:

Cache slot size is 32 bytes = 25 bytes

217 # cache slots available in physical cache = Cache size bytes/bytes per cache block = 222 / 25 = 217

215 # of 4- way set associate cache sets in physical cache = 217 / 22 = 215 cache sets in physical cache (set size = 15 bits)

Set size = 22 cache slots per set = 25 bytes per slot \* 22 cache slots per set = 27 bytes per set.

tag size = 12-bits ) (32 – 15 – 5)

32-bit address

|  |  |  |
| --- | --- | --- |
| Tag = 12-bits | Set = 15-bits | Offset 5-bits |

1. (8 pts) Suppose a byte-addressable computer using 4-way set associative cache has 216 words of main memory (where each word is 32 bits) and a cache of 32 blocks, where each block is 4 words. Show the main memory address format for this machine.

(Hint: Because this architecture is byte-addressable, and the number of addresses is critical in determining the address format, you must convert everything to bytes.)

Answer:

Page 10 Last Updated: March 2018

If main memory is 216 words, then it is 216x22=218 bytes, so there are 18 bits in a main memory address. Each block is 4 words, or 22x22=24 bytes. Therefore, the offset field must have 4 bits. There are 25/22=23 sets, so the set field has 3 bits. That leaves 11 bits for the tag.

18-bit Main Memory address

|  |  |  |
| --- | --- | --- |
| Tag = 11-bits | Set = 3-bits | Offset 4-bits |

1. (8 pts) Assume a direct-mapped cache that holds 4096 bytes, where each block is 16 bytes. Assuming an address is 32 bits and that cache is initially empty, complete the table below. (You should use hexadecimal numbers for all answers.) Which, if any of the addresses will cause a collision (forcing the block that was just brought in to be overwritten) if they are accessed one right after the other?

Ans.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| Address | TAG | Cache location (block) | Offset within block |
| 0x0FF0FABA |  |  |  |
| 0x00000011 |  |  |  |
| 0x0FFFFFFE |  |  |  |
| 0x23456719 |  |  |  |
| 0xCAFEBABE |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Address | TAG | Cache location (block) | Offset within block |
| 0x0FF0FABA | 0x0FF0F | 0xAB | 0xA |
| 0x00000011 | 0x00000 | 0x01 | 0x1 |
| 0x0FFFFFFE | 0x0FFFF | 0xFF | 0xE |
| 0x23456719 | 0x23456 | 0x71 | 0x9 |
| 0xCAFEBABE | 0xCAFEB | 0xAB | 0xE |

The first and last addresses both map to cache block 0xAB and therefore will cause a collision.

1. (8 pts) Memory L1 & L2 Cache, … Ch 7.5 (267…275)

Define the following terms:

1. Locality Principle

Typically 90% of the execution time of the program is spent in just 10% on the code.

1. Temporal locality

When a program references a memory location it is likely to reference that same memory location again soon

1. Spatial locality

A memory location that is near a recently referenced location is more likely to be referenced than a memory location that is further away.

1. Cache Valid Bit

A cache valid bit indicates whether or not the slot holds a line that belongs to the executing program.

1. Cache Dirty Bit

Cache Dirty bit keeps track of cache lines that have been modified while it is in the cache. A dirty cache line must be written back to the main memory before the cache slot is reused for another line.

1. Cache Tag

Tracks which blocks is in each cache slot, a xx-bit tag field, where xx-bits is based on the number and size of blocks cache divides main memory.

1. Fully Associative cache mapping

In an associative mapped cache, each main memory block can be mapped to any cache slot. The mapping from main memory blocks to cache slots is performed by partitioning an address into fields for the tag and the word. The 32-bit main memory is partitioned into a 27 bit field tag followed by a 5 bit word field.

1. Direct-Mapped Cache mapping

For a Direct –Mapped cache, each main memory block can be mapped to only one slot, but each slot can receive more than one block. The mapping from main memory blocks to cache slots is performed by partitioning an address into fields for the tag, the slot, and the word. The 32-bit main memory is partitioned into a 13 bit field tag followed by a 14 bit slot field, followed by a 5 bit word field.

1. Set Associative cache mapping

Set Associative Mapping scheme combines the simplicity of direct mapping with the flexibility of associative mapping. When an address is mapped to a set, the direct mapping scheme is used, and then associative mapping is used within a set. The format for an address has 13 bits in the set field, which identifies the set in which the addressed word will be found if it is in the cache. The 32-bit main memory is partitioned into a 14 -bit field tag followed by a 13 bit slot field, followed by a 5 bit word field.

1. LRU Replacement Algorithm

Least Recently Used, a time stamp is added which is updated when any slot is accessed. When full the least recently used slot is replaced.

1. LFU Replacement Algorithm

Least Frequently Used, a frequency counter is added which is updated when any slot is accessed. When the cache is full the least frequently used slot is replaced.

1. Random Replacement Algorithm

The Random replacement simply replaces a slot to be replaced at random when the cache is full.

1. Write Through Policy

For write operations, there may be two different copies of the word, one in the cache, and one in the main memory. If both are updated simultaneously, this is referred to as ‘Write Through Policy.’ i. e. Write data to both cache and main memory.

1. Write Back Policy

For write operations, there may be two different copies of the word, one in the cache, and one in the main memory. If the write is deferred until the cache line is flushed from the cache, this is referred to as ‘Write Back Policy.’ i.e. Write data to cache only and defer main memory write until block is flushed.

1. Write Allocate

For write operations, there may be two different copies of the word, one in the cache, and one in the main memory. If the data item is not in the cache when the write occurs, there is the choice of bringing the block containing the word into the cache and then updating it, known as ‘Write Allocate.’

1. Write No-Allocate

For write operations, there may be two different copies of the word, one in the cache, and one in the main memory. If the data item is not in the cache when the write occurs, then updating the block containing the word in main memory only without involving the cache, known as ‘Write No-Allocate.’

1. (10 pts) This question tests your knowledge of Virtual Memory Translation tables. This virtual memory system has a page size of 2048 words, 16 virtual pages, four physical page frames, and uses the LFU page replacement policy. Physical Memory is 32KBs (215 bytes). The page table is:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Virtual Memory  Byte Address | Virtual Memory  Word Address | Page Virtual  Address | Present  Bit | Disk  Address | Page Frame  Field |
| 0000 – 8191 | 0000 - 2047 | 00 | 0 | 0100 1011 100 | xx |
| 8192– 16383 | 2048 – **4095** | 01 | 1 | 1110 1110 010 | 00 |
| 16384 - 24575 | 4096 – 6143 | 02 | 0 | 1011 0010 111 | xx |
| 24576 – 32767 | 6144 – 8191 | 03 | 0 | 0000 1001 111 | xx |
| 32768 - 40959 | 8192 – 10239 | 04 | 0 | 0101 1100 101 | xx |
| 40960 - 49151 | 10240 - **12287** | 05 | 0 | 1010 0111 001 | xx |
| 49152 - 57343 | 12288 - 14335 | 06 | 1 | 0011 0101 100 | 11 |
| 57344 - 65535 | 14336 - 16383 | 07 | 0 | 0101 0001 011 | xx |
| 65536 – 73727 | 16384 – 18431 | 08 | 1 | 0110 1011 100 | 01 |
| 73728 – 81919 | 18432 – 20479 | 09 | 0 | 1110 0110 010 | xx |
| 81920 – 90111 | 20480 – 22527 | 10 | 0 | 1011 1010 111 | xx |
| 90112 – 98303 | 22528 – 24575 | 11 | 0 | 0000 1001 011 | xx |
| 98304 – 106495 | 24576 – 26623 | 12 | 0 | 0111 1100 101 | xx |
| **106496** – 114687 | 26624 – 28671 | 13 | 1 | 1011 0111 001 | 10 |
| 114688 – 122879 | 28672 – 30719 | 14 | 0 | 0011 1101 100 | xx |
| 122880 – 131071 | 30720 - 32767 | 15 | 0 | 0101 1001 011 | xx |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Physical Memory  Byte Address | Physical Memory  Word Address | Page Frame  Address | Page Frame  Field | LFU Counter |
| 0000 – 8191 | 0000 - **2047** | 00 | 00 | 05 |
| 8192– 16383 | 2048 – 4095 | 01 | 01 | 15 |
| **16384** - 24575 | 4096 – **6143** | 02 | 10 | 02 |
| 24576 – 32767 | 6144 – 8191 | 03 | 11 | 07 |

* 1. What is the physical memory **word address** for the virtual word address **4095**?

Answer: main memory address 2047

* 1. What is the physical memory byte address for virtual byte address **106496**?

Answer: Page Frame Field is ‘02’, physical memory byte address 16384

* 1. A fault occurs on **virtual page address 00**. Which physical page frame will be replaced and used for virtual page 0?

Answer: Page frame 2; Page Frame Field ‘10’.

* 1. What is the physical memory **word address** for the virtual word address **12287**?

Answer: Word not in memory… therefor replacement page for LFU is Page Frame 02 providing main memory address 6143