MICROPROCESSOR AND COMPUTER ARCHITECTURE

CACHE MEMORY

Mapping Techniques
Problems

Prof. Roopa Ravish

Example 1

Consider a direct mapped cache of size 512 KB with block size 1 KB. There are 7 bits in the tag. Find the Size of main memory

Given:

Cache memory size = 512 KB

Block size = Line size = 1 KB = 2^10, i.e offset is 10 bits

Number of bits in tag = 7 bits

Total number of blocks= 512 kb / 1 kb = 512 blocks or line = 2^9 . i.e 9 bits

The physical address = TAG+ Line number + offset

= 7 bits + 9 bits + 10 bits

= 26 bits

Size of memory is 2^26= 67,108 bytes

Example 2

A computer system uses 16-bit memory addresses. It has a 2K-byte cache organized in a direct-mapped manner with 64 bytes per cache block. Assume that the size of each memory word is 1 byte.

(a) Calculate the number of bits in each of the Tag, Block, and Word fields of the memory address.

(b) When a program is executed, the processor reads data sequentially from the following word addresses: **128, 144, 2176, 2180, 128, 2176**

All the above addresses are shown in decimal values. Assume that the cache is initially empty.

For each of the above addresses, indicate whether the cache access will result in a hit or a miss.

Example 2

Block size = 64 bytes = 2^6 bytes
Therefore, **Number of bits in the *Word* field = 6**
Cache size = 2K-byte = 2^{11} bytes
Number of cache blocks = Cache size / Block size = $2^{11}/2^6 = 2^5$
Therefore, **Number of bits in the *Block* field = 5**
Total number of address bits = 16
Therefore, **Number of bits in the *Tag* field = 16 - 6 - 5 = 5**
For a given 16-bit address, the 5 most significant bits, represent the *Tag*, the next 5 bits represent the *Block*, and the 6 least significant bits represent the *Word*.

Example 2

Access # 1: The cache is initially empty. Therefore, all the cache blocks are invalid

Address = $(128)_{10} = (0000000010000000)_2$
For this address, *Tag* = 00000, *Block* = 00010, *Word* = 000000

Since the cache is empty before this access, this will be a cache **miss**
After this access, **Tag field for cache block 00010 is set to 00000**

Access # 2: Address = $(144)_{10} = (0000000010010000)_2$

For this address, *Tag* = **00000**, *Block* = **00010**, *Word* = 010000

Since tag field for cache block **00010** is **00000** before this access, this will be a cache **hit** (because address tag = block tag)

Example 2

Access # 3: Address = $(2176)_{10}$ = ($\mathbf{00001}\mathbf{00010}00000)_2$
For this address, *Tag* = **00001**, *Block* = **00010**, *Word* = 000000
Since tag field for cache block **00010** is **00000** before this access, this will be a cache **miss**
(address tag ≠ block tag)
After this access, **Tag field for cache block 00010 is set to 00001**

Access # 4:
Address = $(2180)_{10}$ = ($\mathbf{00001}000100100)_2$
For this address, *Tag* = **00001**, *Block* = **00010**, *Word* = 000100
Since tag field for cache block **00010** is **00001** before this access, this will be a cache **hit** (address tag = block tag)

Example 2

Access # 5:
Address = $(128)_{10}$ = ($\mathbf{00000}00010\mathbf{0}0000)_2$
For this address, *Tag* = **00000**, *Block* = **00010**, *Word* = 000000
Since tag field for cache block **00010** is **00001** before this access, this will be a cache **miss**
(address tag ≠ block tag)
After this access, **Tag field for cache block 00010 is set to 00000**

Access # 6:
Address = $(2176)_{10}$ = ($\mathbf{00001}00010\mathbf{0}0000)_2$
For this address, *Tag* = **00001**, *Block* = **00010**, *Word* = 000000
Since tag field for cache block **00010** is **00000** before this access, this will be a cache **miss**
(address tag ≠ block tag)
After this access, **Tag field for cache block 00010 is set to 00001**

Cache hit rate = Number of hits / Number of accesses = 2/6
= 0.333

Example 3

Consider a 4 way set associative cache with block size 4 KB. The size of main memory is 16 GB and there are 10 bits in the tag. Find the Size of cache memory

Given-

- Block size = Line size = 4 KB = 2^12 i.e. offset = 12 bits
- Size of main memory = 16 GB = 2^34 bytes i.e Physical address = 34 bits
- Number of bits in tag = 10 bits

- Set number = 34- (10+12)=12 bits

- Number of blocks or lines = 2^12 x 4 = 2^14 lines

- Size of Cache= 2^14 * 4 kb = 2^16 kb= 64 MB

Example 4

A computer system uses 16-bit memory addresses. It has a 2K-byte cache organized in a 2-way set associative manner with 64 bytes per cache block. Assume that the size of each memory word is 1 byte.

(a) Calculate the number of bits in each of the Tag, Block, and Word fields of the memory address.

(b) When a program is executed, the processor reads data sequentially from the following word addresses: **128, 144, 2176, 2180, 128, 2176**

All the above addresses are shown in decimal values. Assume that the cache is initially empty.

For each of the above addresses, indicate whether the cache access will result in a hit or a miss.

Example 4

Block size = 64 bytes = 2^6 bytes
Therefore, **Number of bits in the *Word* field = 6**
Cache size = 2K-byte = 2^{11} bytes
Number of cache blocks = Cache size / (Block size x Number of blocks per set)
$= 2^{11}/(2^6 \times 2) = 2^4$
Therefore, **Number of bits in the *set* field = 4**
Total number of address bits = 16
Therefore, **Number of bits in the *Tag* field = 16 – (6+4) = 6**
For a given 16-bit address, the 6 most significant bits, represent the *Tag*, the next 4 bits represent the *Block*, and the 6 least significant bits represent the *Word*.

Example 4

Access # 1:
Address = $(128)_{10}$ = $(0000000010000000)_2$
For this address, *Tag* = **000000**, *Set* = **0010**, *Word* = 000000
Since the cache is empty before this access, this will be a cache **miss**
After this access, **Tag field for the first block in set 0010 is set to 000000**

Access # 2:
Address = $(144)_{10}$ = $(0000000010010000)_2$
For this address, *Tag* = **000000**, *Set* = **0010**, *Word* = 010000
The tag field for this address matches the tag field for the first block in set 0010.
Therefore, this access will be a cache **hit**.

Example 4

Access # 3:
Address = $(2176)_{10}$ = $(000010\,0010\,000000)_2$
For this address, *Tag* = **000010**, *Set*= **0010**, *Word* = 000000
The tag field for this address does not match the tag field for the first block in set 0010.

The second block in set 0010 is empty. Therefore, this access will be a cache **miss**.
After this access, **Tag field for the second block in set 0010 is set to 000010**

Access # 4:
Address = $(2180)_{10}$ = $(000010\,0010\,000100)_2$
For this address, *Tag* = **000010**, *Set* = **0010**, *Word* = 000100
The tag field for this address matches the tag field for the second block in set 0010.
Therefore, this access will be a cache **hit**.

Example 4

Access # 5:
Address = $(128)_{10}$ = $(0000000010000000)_2$
For this address, *Tag* = **000000**, *Set* = **0010**, *Word* = 000000
The tag field for this address matches the tag field for the first block in set 0010.
Therefore, this access will be a cache **hit**.

Access # 6:
Address = $(2176)_{10}$ = $(0000100010000000)_2$
For this address, *Tag* = **000010**, *Set* = **0010**, *Word* = 000000
The tag field for this address matches the tag field for the second block in set 0010.
Therefore, this access will be a cache **hit**.

Example 4

Cache hit rate = Number of hits / Number of accesses = 4/6 = 0.666

Compare with direct mapping

Cache hit rate = Number of hits / Number of accesses
= 2/6
= 0.333

Example 5

5. Consider a 4-way set associative cache consisting of 128 lines with a line size of 64 words. The CPU generates a 20-bit address of a word in main memory. The numbers of bits in the TAG, LINE and WORD fields are _____

Consider a 4-way set associative cache consisting of 128 lines with a line size of 64 words. The CPU generates a 20-bit address of a word in main memory. The numbers of bits in the TAG, LINE and WORD fields are _____

Line/block size= 64 words
Main memory address bits= 20
Number of sets in the cache = 128/4= 32 = 2^5
Word offset= 64= 2^6
Set bits=5
Word bits=6
Tag=20-(5+6)= 9

Example 6

6. A processor can support a maximum memory of 4 GB, where the memory is word-addressable (a word consists of two bytes). The size of the address bus of the processor is at least _____ bits.

A processor can support a maximum memory of 4 GB, where the memory is word-addressable (a word consists of two bytes). The size of the address bus of the processor is at least _____ bits.

31 bits.

Example 7

7. A certain processor uses a fully associative cache of size 16 kB. The cache block size is 16 bytes. Assume that the main memory is byte addressable and uses a 32-bit address. How many bits are required for the Tag and the Index fields respectively in the addresses generated by the processor?

A certain processor uses a fully associative cache of size 16 kB. The cache block size is 16 bytes. Assume that the main memory is byte addressable and uses a 32-bit address. How many bits are required for the Tag and the Index fields respectively in the addresses generated by the processor?

Main memory address = 32 bits
In fully associative cache main memory address format there is no index bits.
∴ Tag bits = Total bits - Offset bits = 32 - 4 = 28

So, tag bits = 28, Index bits = 0

Example 8

8. A block-set associative cache memory consists of 128 blocks divided into four block sets. The main memory consists of 16,384blocks and each blocks contains 256eight bit words.

(i) How many bits are required for addressing the main memory?
(ii) How many bits are needed to represent the TAG, SET and WORD fields?

A block-set associative cache memory consists of 128 blocks divided into four block sets. The main memory consists of 16,384blocks and each blocks contains 256eight bit words.

(i) How many bits are required for addressing the main memory?
(ii) How many bits are needed to represent the TAG, SET and WORD fields?

Ans: cache =128 blocks
4 block/set, 128/4 = 32 sets = 2^5
Block- 256 words = 2^8
Main memory address= 16K X 256 =2^4 X 2^10 X 2^8= 2^22
= 22 bits
Tag = 22-(5+8) = 9
(i) 22 bits (ii) TAG =9bits SET =5bits WORD =8bits

Example 9

9. Consider a small two-way set-associative cache memory, consisting of four blocks. For choosing the block to be replaced, uses the least recently used (LRU) scheme. The number of cache misses for the following sequence of blocks addresses is 8,12,0,12,8

Consider a small two-way set-associative cache memory, consisting of four blocks. For choosing the block to be replaced, uses the least recently used (LRU) scheme. The number of cache misses for the following sequence of blocks addresses is 8,12,0,12,8

4

Example 10

10. Consider a direct mapped cache of size 32KB with block size 32 bytes. The CPU generates 32 bit addresses. The number of bits needed for cache indexing and the number of tag bits are respectively _____

Consider a direct mapped cache of size 32KB with block size 32 bytes. The CPU generates 32 bit addresses. The number of bits needed for cache indexing and the number of tag bits are respectively _____

15 and 17

Example 11

11. A CPU has 32-bit memory address and a 256KB cache memory. The cache is organized as a 4-way set associative cache with cache block size of 16 bytes.
(a) What is the number of sets in the cache?
(b) What is the size (in bits) of the tag field per cache block?
(c) What is the number and size of comparators required for tag matching?
(d) How many address bits are required to find the byte offset within a cache block?
(e) What is the total amount of extra memory (in bytes) required for the tag bits?

A CPU has 32-bit memory address and a 256KB cache memory. The cache is organized as a 4-way set associative cache with cache block size of 16 bytes.

(a) What is the number of sets in the cache?

(b) What is the size (in bits) of the tag field per cache block?

(c) What is the number and size of comparators required for tag matching?

(d) How many address bits are required to find the byte offset within a cache block?

(a) 4K Sets
(b) Size of the tag field =16
(c) 4 comparators required and each size is 16bits
(d) 4bits