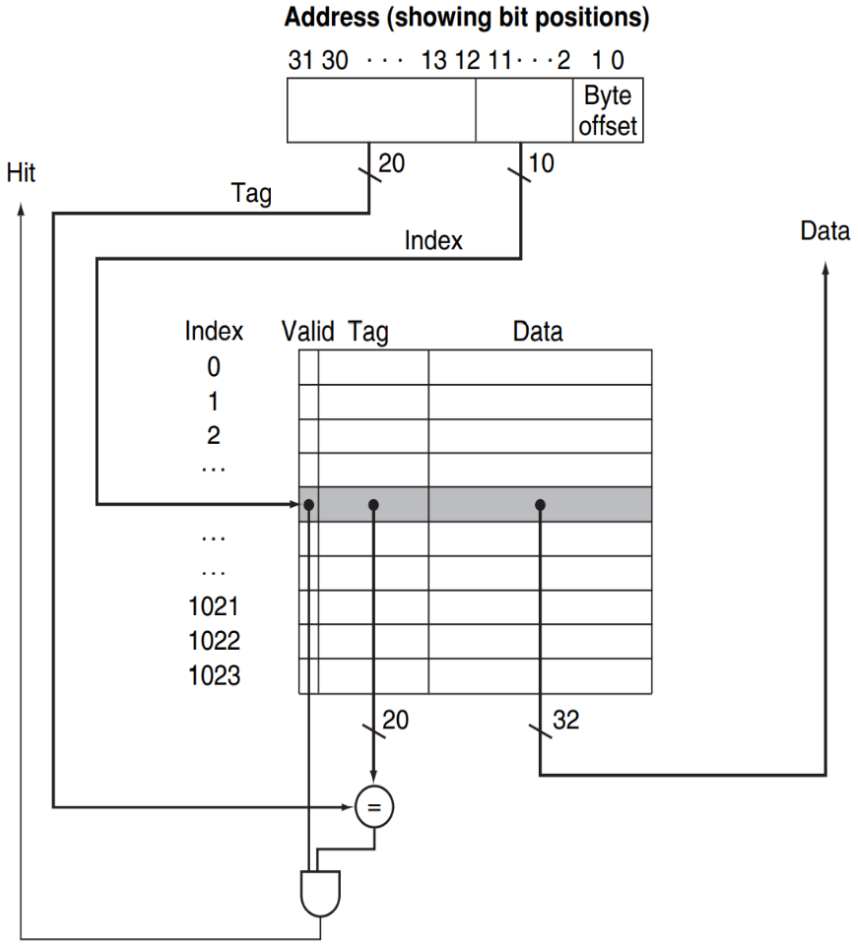
Computer Architecture (ENE1004)

▼ Lec 19

Lec 19: Large and Fast : Exploting Memory Hierarchy 3

Direct Mapped Cache for the Real World (1)



- Assumption (가정)
  - 32-bit address (vs 5-bit address) (32비트 주소 (vs 5비트 주소))
  - The cache size is 2^10 blocks (vs 8 blocks) (캐시 크기는 2^10 블록 (vs 8 블록))

    - The block size is 2^2 bytes (블록 크기는 2^2바이트이다)
    - 1바이트 = 8비트
- Now, let us determine how an address of data is used for a cache access (이제 캐시 엑세스에 데이터 주소가 어떻게 사용되는지 알아보자)
- Byte offset (바이트 오프셋)
    - Each cache block can contain 2^2 bytes of data (각 캐시 블록은 2^2바이트의 데이터를 포함할 수 있다)
    - The base unit of a data item is a "byte" (데이터 항목의 기본 단위는 "바이트"이다)
    - We need to identify a byte of data within cache block (캐시 블록 내에서 데이터의 바이트를 식별해야 한다)
    - The last 2 bits are used for byte offset (마지막 2비트는 바이트 오프셋에 사용된다)
    - A byte of data is referenced, 2^2 bytes of data including the referenced and neighboring data are brought from the memory to the cache (1바이트의 데이터가 참조되고, 참조된 데이터와 인접한 데이터를 포함한 2^2바이트의 데이터가 메모리에서 캐시로 가져온다)
    - This behavior allows a cache to take advantage of "spatial locality" (이 동작을 통해 캐시는 "공간적 지역성"을 활용할 수 있다)

Direct Mapped Cache for the Real World (2)

- Assumption (가정)
    - 32-bit address (vs 5-bit address) (32비트 주소 (vs 5비트 주소))
    - The cache size is 2^10 blocks (vs 8 blocks) (캐시 크기는 2^10 블록 (vs 8 블록))
    - The block size is 2^2 bytes (블록 크기는 2^2바이트이다)
    - 1바이트 = 8비트
- Index (to select a cache block) (인덱스 (캐시 블록을 선택하기 위해))
    - There are 2^10 blocks in the cache (캐시에는 2^10개의 블록이 있다)
    - We need "10 bits" to determine the index of the data (데이터의 인덱스를 결정하려면 "10비트"가 필요하다)

- Tag (to identify a specific data item) (태그 (특정 데이터 항목을 식별하기 위해))
  - The remaining “20 bits” are used for the tag (나머지 “20 비트”는 태그에 사용된다)
  - 20 = 32 – 10 (for index) – 2 (for byte offset)
- Cache access (캐시 엑세스)
  - Given an address, the target cache block is accessed using the 10-bit (b11-b2) index value (주소가 주어지면 10비트(b11-b2) 인덱스 값을 사용하여 대상 캐시 블록에 엑세스한다)
  - Then, the 20-bit (b31-b12) tag value is compared to the tag field of the cache block (그런 다음 20비트(b31-b12) 인덱스 값을 사용하여 대상 캐시 블록에 엑세스한다)
  - If they are equal (“hit”), the entire 4-byte data is read (둘이 같으면(”히트), 4바이트 데이터 전체를 읽는다)
  - If they do not equal (“miss”), the referenced 4-byte data replaces the existing 4-byte data in the cache (같지 않다면(”미스”), 참조된 4바이트 데이터가 캐시의 기존 4바이트 데이터를 대체한다)

Generalization of Direct Mapped Cache (1)

- Assumption (가정)
  - 32-bit address (32비트 주소)
  - The cache size is 2^n blocks (캐시 크기는 2^n 블록)
  - The block size is 2^m words (블록 크기는 2^(m+2)바이트이다)
  - 1바이트 = 8비트
- Byte offset (바이트 오프셋)
  - Each cache block can contain 2^(m+2) bytes of data (각 캐시 블록은 2^(m+2)바이트의 데이터를 포함할 수 있다)
  - “m bits” are used to identify a word within a block (”m 비트”는 블록 내의 word를 식별하는 데 사용된다)
  - “2 bits” are used to identify a byte within a word (”2 비트”는 word 내의 바이트를 식별하는 데 사용된다)

    - The last m+2 bits are used for byte offset (마지막 m+2 비트는 바이트 오프셋에 사용된다)
- Index (to select a cache block) (인덱스 (캐시 블록을 선택하기 위해))
    - There are 2^n blocks in the cache (캐시에는 2^n개의 블록이 있다)
    - We need "n bits" to determine the index of the data (데이터의 인덱스를 결정하려면 "n비트"가 필요하다)
- Tag (to identify a specific data item) (태그 (특정 데이터 항목을 식별하기 위해))
    - The remaining bits are used for the tag (나머지 비트는 태그에 사용된다)
    - The size of the tag filed is 32 – (n + m + 2)

Generalization of Direct Mapped Cache (2)

- Assumption (가정)
    - 32-bit address (32비트 주소)
    - The cache size is 2^n blocks (캐시 크기는 2^n 블록)
    - The block size is 2^m words (블록 크기는 2^(m+2)바이트이다)
    - 1바이트 = 8비트
- Address fields
    - Byte offset: m+2 bits
    - Index: n bits
    - Tag: 32 – (n + m + 2) bits
- The total number of bits in a cache (캐시의 총 비트 수)
    - # of blocks x (block size + tag size + valid bit size) (블록 개수 * (블록 크기 + 태그 크기 + 유효 비트 크기))
    - 2^n x (2^(m+2) bytes + (32-n-m-2) bits + 1 bit)
    - 2^n x (2^(m+5) + (32-n-m-2) + 1) bits
- Due to the tag and valid bit fields, the actual size of the cache is larger than the total amount of data (2^n x 2^(m+2) bytes) (태그와 유효한 비트 필드로 인해 캐시의 실제 크기는 총 데이터 양 (2^n x 2^(m+2) bytes)보다 크다)

Bits in Cache (캐시 내 비트)

- Question: How many total bits are required for a direct mapped cache with 16 KiB of data and 4-word blocks, assuming a 32-bit address? (질문: 32비트 주소가 있다고 가정할 때 16KiB의 데이터와 4 word 블록이 있는 직접 매핑된 캐시에 필요한 총 비트는 몇 개인가?)
    - 1 Kib = 2^10 bytes = 1,024 bytes
    - 1 word = 4 bytes
- Hint: We need to calculate “total # of blocks (indices)” and “# bits for tag” (힌트 : “총 블록(인덱스) 개수”와 “태그의 비트 수”를 계산해야 한다)
    - Cache size = total # blocks x (block size + tag size + valid bit size) for each block (캐시 크기 = 총 블록 수 x 각 블록의 (블록 크기 + 태그 크기 + 유효 비트 크기))
- The total number of cache blocks (총 캐시 블록 수)
    - The entire size of blocks = 16 KiB (2^14 Bytes = 2^12 words) (전체 블록 크기 = 16Kib (2^14 바이트 = 2^12 워드))
    - The size of each block = 4 words (2^4 Bytes = 2^2 words) (각 블록의 크기 = 4 워드 (2^4바이트 = 2^2워드))
    - The total number of blocks = entire size / a block size = 2^10 = 1024 (총 블록 수 = 전체 블록 크기 / 각 블록의 크기 = 2^10 = 1024)
    - The number of bits for “index” is 10 bits (”인덱스”의 비트 수는 10비트이다)
- The number of bits for “tag” (”태그”의 비트 수)
    - # bits for tag = total # address bits – (# bits for index + # bits for byte offset) (태그의 비트 수 = 총 주소 비트 수 - (인덱스의 비트 수 + 바이트 오프셋의 비트 수))
    - # bits for byte offset = 4, because the size of a cache block is of 4-word (2^4 Bytes) (캐시 블록의 크기가 4워드(2^4 바이트)이므로 바이트 오프셋의 비트 수 = 4이다)
    - # bits for tag = 32 – (10 + 4) = 18 (태그의 비트 수 = 32 - (10 + 4) = 18)
- Cache size = total # blocks x (block size + tag bit size + valid bit size) for each block (캐시 크기 = 총 블록 개수 x 각 블록의 (블록 크기 + 태그 비트 크기 + 유효 비트 크기))

- Cache size = 1024 x (4 words + 18 bits + 1 bit) = 1024 x (2^7 bits + 18 bits + 1 bit) = 2^10 × 147 bits
- 147 Kbits

• 32 비트 주소가 있다고 가정할 때 16 KiB의 데이터와 4 word 블록이 있는 총 비트는 몇 개인가?

- 32 비트 주소 —— 16 KiB = 2^{14} bytes 의 데이터
  4 word = 2^4 bytes 의 블록

• 힌트: 총 블록 (인덱스) 개수와 태그의 비트 수를 계산해야한다.

- 캐시 크기 = 총 블록수 x 각 블록의 (블록 크기 + 태그 크기 + 유효 비트 크기)

• 총 캐시 블록 수

- 전체 블록 크기 = 16 KiB = 2^{14} bytes
- 각 블록의 크기 = 4 word = 2^4 bytes
- 총 블록 수 = $\frac{\text{전체 블록 크기 (캐시 크기)}}{\text{각 블록의 크기}} = \frac{2^{14}\ \text{bytes}}{2^4\ \text{bytes}} = 2^{10} = 1024$
- 인덱스의 비트 수는 10비트이다.

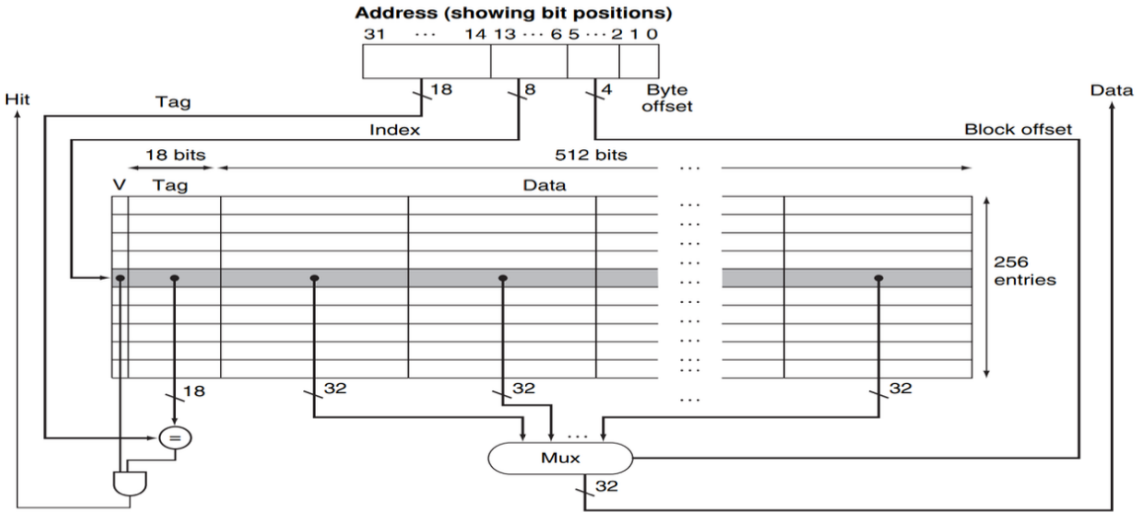
• 태그의 비트 수

- 태그의 비트 수 = 총 주소 비트 수 - (인덱스의 비트 수 + 바이트 오프셋의 비트 수)
- 캐시 블록의 크기가 4 word (2^4 bytes) 이므로 바이트 오프셋의 비트 수 = 4이다)
- 태그의 비트 수 = 32 - (10+4) = 18

• 캐시 크기 = 총 블록수 x 각 블록의 (블록 크기 + 태그 크기 + 유효 비트 크기)

- 캐시 크기 = 2^{10} x (4 word + 18 bits + 1 bit) = 2^{10} x (2^7 bits + 18 bits + 1 bit) = 2^{10} x 147 bits = 147 Kbits

Bits in Cache (2)



- Cache block size: 512 bits = 64 (2^6) Bytes = 16 words (캐시 블록 크기 : 512 비트 = 64 (2^6) 바이트 = 16(2^4) 워드)
    - Byte offset = 6 bits (b5-b0)
    - You may want to access the data in the unit of “word”; then, you can use upper bits in the byte offset bits; in this example, upper 4 bits (b5-b2) is used for word offset (called “block offset”) (“워드” 단위로 데이터에 액세스하려면 바이트 오프셋 비트에서 상위 비트를 사용할 수 있으며, 이 예에서는 상위 4비트(b5-b2)가 워드 오프셋에 사용된다(“블록 오프셋”이라고 함).)
- Total number of cache blocks is 256 (2^8); # bits for index is 8 (b13-b6) (캐시 블록의 총 개수는 256(2^8)이며, 인덱스의 비트 수는 8 (b13-b6)이다.)
- # bits for tag = 32 – (8 + 6) = 18 bits (태그의 비트 수 = 32 - (8 + 6) = 18 비트)
- The cache size = 256 x (512 bits for data + 18 bits for tag + 1 bit for valid bit) (캐시 크기 = 256 x (데이터를 위한 블록 크기 512비트 + 태그 18비트 + 유효 비트 1비트)