# CS147 - Lecture 20

Kaushik Patra
(kaushik.patra@sjsu.edu)

1

- Cache Structure
- Cache Operation
- Cache Circuit

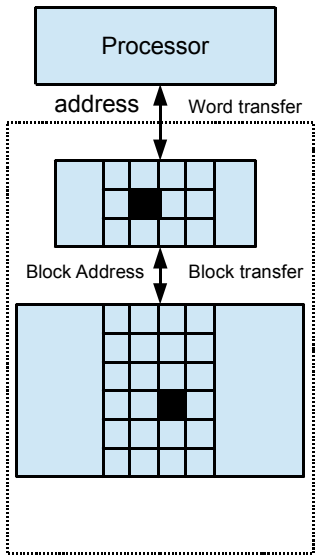
*Reference Books / Source:*

1) *Chapter 4 of 'Computer Organization & Architecture' by Stallings*
2) *Chapter 7 of 'Computer Organization & Design' by Patterson and Hennessy/*

Cache Memory Structure ...

2

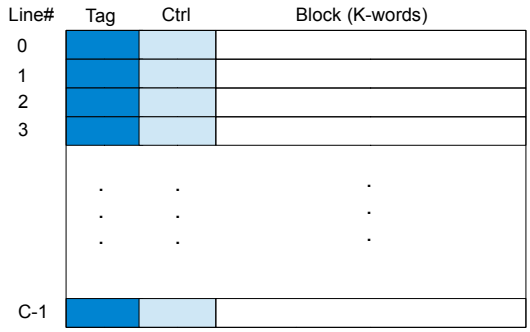
# Cache – Structure



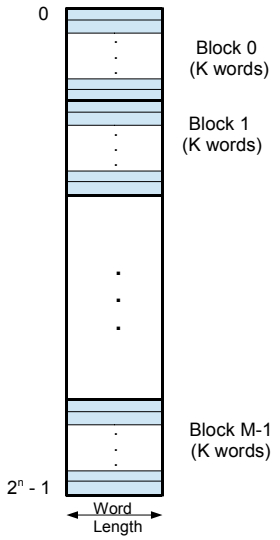

- Key observations
  - Upper level memories are faster.
  - Lower level memories are larger
  - Information transfer happens only between two levels of the memories adjacent to each other in the memory hierarchy.
  - Information transfer happens in block (not a single word).
  - Memory hierarchy is transparent to processor. Processor issues address to memory system assuming it is a single big and fast memory system.

3

# Cache – Structure




Cache Memory




Main Memory

- Memory is divided into M blocks of equal size of K words.
- Cache memory is organized into C lines (Cache line)
  - Each line holds K words (or an block)
  - C < M; thus multiple block can be mapped into same cache line.
- Each line is associated with tag and control bits.
  - Tag is to determine if a block is present in the target line or not.
  - Control bit is to represent various states (like valid, dirty, etc.) of the cache line.

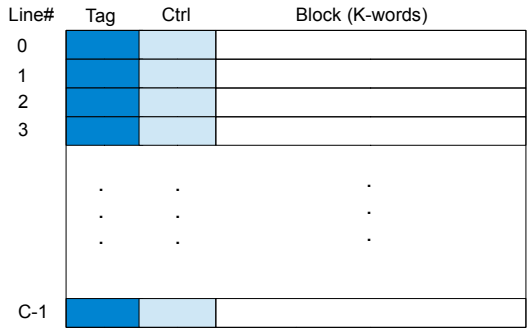
4

- The entire addressable space of the main memory is divided into equal length M region (called block) with K words. This blocks are the unit of memory to be transferred between cache memory and main memory.

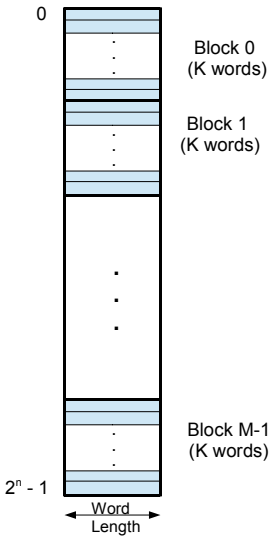
- The cache memory is organized into C cache lines (indexed or addressed from 0 to C-1). Each cache lines contains K words, a tag field (which works as analogous to hash key in hash table data structure) and a control filed (to define status of the cache line). Number of cache line is usually much less that the number of blocks in the main memory (C < M).

  - The tag determines if a block is present on a cache line or not.
  - Control bit determines status of a cache line (valid, dirty, etc. - to be discussed later).

# Cache – Structure




Cache Memory




Main Memory

- Block number B of an reference address A is the block address of the reference address (B = A div K).
- Line index I of a block address B is modulo of B with C (I = B mod C).
- For each A, an unique tag (T) has to be created.
  - T = B div C

5

• Cache operation is always described in terms of the block number or block address in the main memory (because unit of information transfer between cache and the memory is always the block). A block of K words) corresponds to total information of K neighboring addresses. To translate a memory address A into it corresponding block address B we integer divide A by K and take the quotient term. This means multiple addresses (exactly K number of addresses) are mapped into one block address (many to one mapping).

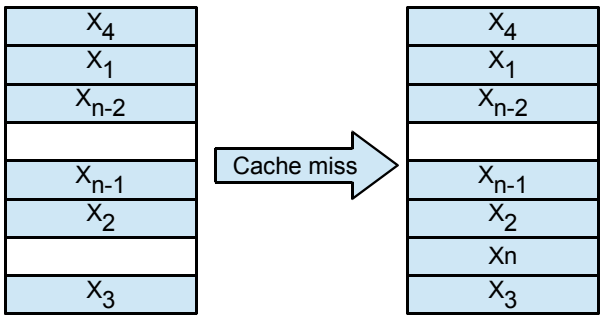
• One of the technique to map a block address B to corresponding cache line index I is to take modulo of B with respect to C – means we do a integer divide B by C and take the remainder of the division operation. This guarantees that the computed cache line index will stay within 0 .. (C-1) range. This specific technique is direct mapping technique. In this mapping one cache line can be mapped from multiple block addresses (many to one mapping).

• A tag T for a block address is created by integer division of B by C and taking the quotient. A (I,T) pair is unique for a given address A.

Cache Memory Operation ...

6

## Cache – Operation



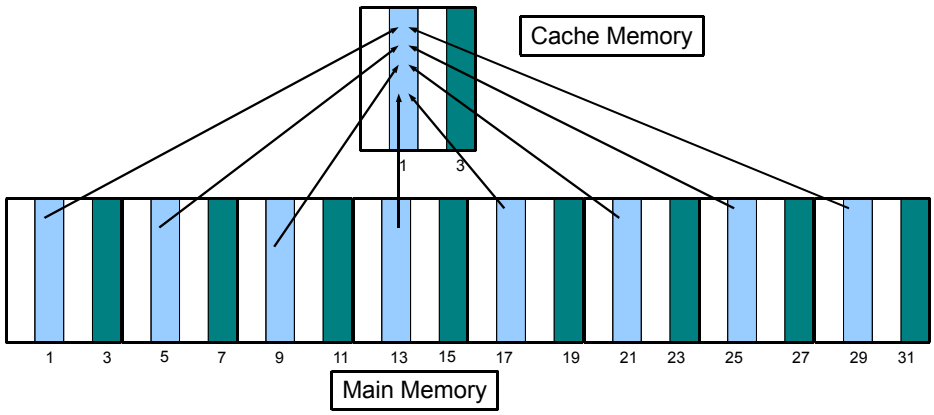

- If a block $X_n$ is referenced, but not present in cache then this block will be copied from lower level memory.
- Two questions
  - How do we know if an block is already at cache?
  - If present, where to find the block?

7

- Without knowing any details on the cache structure, what we can infer on the cache operation is that if an address A is referenced the processor then if corresponding block $X_n$ is not present in the cache it will be brought into the cache by block transfer from lower level memory and then the information corresponding to the address A will be returned back to the processor (similar to write , but a little bit different though).

- There are two problems to resolve. First is how to know if a block is present in the cache. Secondly, where to find the block in the cache if present. However, these two questions are interrelated.

# Cache – Operation




- If a block can go into only a specific cache line, then it is easy to find the block if present.
  - Cache index formula I = (B mod C)
  - Gives one specific cache line index – solves where to find a block.
  - Issue - multiple block may have the same index line
    - Use TAG to test is the desired block is present or not.

8

- To answer the second question on the previous slide the cache index formula is enough to find a block in the cache. Any block address will be mapped into only one fixed cache line index I by this modulo operation. Now the problem becomes that how to test if required block is present in the cache line or not. Multiple block may have the same cache line index.Without any test, we can not be sure if the required block is present in that cache line. To resolve this problem, the tag bit pattern is used. The tag T of a block address B is computed and stored as a part of a cache line in such a way that (I,T) will be unique for a given address A (B is computed from A). Hence if we compute tag T of referenced address A and match it to the stored tag value in the cache line with index I, we can conclude whether the required block is present in the cache or not.

# Cache – Operation

OPERATIONS

| | BLOCK ADDR | CACHE LINE | TAG | HIT / MISS |
|---|---|---|---|---|
| 22 | 10110 | 110 | 10 | M |
| 26 | 11010 | 010 | 11 | M |
| 22 | 10110 | 110 | 10 | H |
| 26 | 11010 | 010 | 11 | H |
| 16 | 10000 | 000 | 10 | |
| 03 | 00011 | 011 | 00 | |
| 16 | 10000 | 000 | 10 | |
| 18 | 10010 | 010 | 10 | |

CACHE CONTENT - INITAL

| CACHE LINE | V | TAG | DATA |
|---|---|---|---|
| 000 | N | | |
| 001 | N | | |
| 010 | N | | |
| 011 | N | | |
| 100 | N | | |
| 101 | N | | |
| 110 | N | | |
| 111 | N | | |

CACHE CONTENT – OP1

| CACHE LINE | V | TAG | DATA |
|---|---|---|---|
| 000 | N | | |
| 001 | N | | |
| 010 | N | | |
| 011 | N | | |
| 100 | N | | |
| 101 | N | | |
| 110 | Y | 10 | BLOCK for 10110 |
| 111 | N | | |

CACHE CONTENT – OP2 / OP3 / OP4

| CACHE LINE | V | TAG | DATA |
|---|---|---|---|
| 000 | N | | |
| 001 | N | | |
| 010 | Y | 11 | BLOCK for 11010 |
| 011 | N | | |
| 100 | N | | |
| 101 | N | | |
| 110 | Y | 10 | BLOCK for 10110 |
| 111 | N | | |

9

- Let's review the cache operation in a very miniature scale. In this example, there are 8 cache lines (indexed with 3 bit value) with 5-bit block address. This means we need to have 2-bit TAG field per cache line. The cache also has a valid bit 'V' which indicates if a cache line is valid (or loaded) or not. Let's assume all the block references are to read information for a corresponding address.

- The operation table shows sequence of 8 block address references and corresponding computed cache line and tag.

- The initial state of cache is the valid bit is all 'N' meaning invalid. The tag and the data fields are empty.

- For operation 1, cache line 6 is referenced which is invalid at the operation. The cache access is a miss in this case. The corresponding block is copied from lower level memory to cache line 6. Once the copy is done, the valid bit is turned to 'Y'and the tag is stored as 2. The corresponding information is returned back to the processor.

- For operation 2, cache line 2 is referenced which is invalid at the operation. The cache access is a miss in this case. The corresponding block is copied from lower level memory to cache line 2. Once the copy is done, the valid bit is turned to 'Y'and the tag is stored as 3. The corresponding information is returned back to the processor.

- For operation 3, cache line 6 is referenced which has a valid bit value 'Y' and tag as same at the required tag value 2. The the cache access is a hit and corresponding information is returned back to the processor.

- For operation 4, cache line 2 is referenced which has a valid bit value 'Y' and tag as same at the required tag value 3. The the cache access is a hit and corresponding information is returned back to the processor.

# Cache – Operation

OPERATIONS

| | BLOCK ADDR | CACHE LINE | TAG | HIT / MISS |
|---|---|---|---|---|
| 22 | 10110 | 110 | 10 | M |
| 26 | 11010 | 010 | 11 | M |
| 22 | 10110 | 110 | 10 | H |
| 26 | 11010 | 010 | 11 | H |
| 16 | 10000 | 000 | 10 | M |
| 03 | 00011 | 011 | 00 | M |
| 16 | 10000 | 000 | 10 | H |
| 18 | 10010 | 010 | 10 | M |

CACHE CONTENT – OP5

| CACHE LINE | V | TAG | DATA |
|---|---|---|---|
| 000 | Y | 10 | BLOCK for 10000 |
| 001 | N | | |
| 010 | Y | 11 | BLOCK for 11010 |
| 011 | N | | |
| 100 | N | | |
| 101 | N | | |
| 110 | Y | 10 | BLOCK for 10110 |
| 111 | N | | |

CACHE CONTENT – OP6

| CACHE LINE | V | TAG | DATA |
|---|---|---|---|
| 000 | Y | 10 | BLOCK for 10000 |
| 001 | N | | |
| 010 | Y | 11 | BLOCK for 11010 |
| 011 | Y | 00 | BLOCK for 00011 |
| 100 | N | | |
| 101 | N | | |
| 110 | Y | 10 | BLOCK for 10110 |
| 111 | N | | |

CACHE CONTENT – OP7 / OP8

| CACHE LINE | V | TAG | DATA |
|---|---|---|---|
| 000 | Y | 10 | BLOCK for 10000 |
| 001 | N | | |
| 010 | Y | 10 | BLOCK for 10010 |
| 011 | Y | 00 | BLOCK for 00011 |
| 100 | N | | |
| 101 | N | | |
| 110 | Y | 10 | BLOCK for 10110 |
| 111 | N | | |

10

- For operation 5, cache line 0 is referenced which is invalid at the operation. The cache access is a miss in this case. The corresponding block is copied from lower level memory to cache line 0. Once the copy is done, the valid bit is turned to 'Y'and the tag is stored as 2. The corresponding information is returned back to the processor.

- For operation 6, cache line 3 is referenced which is invalid at the operation. The cache access is a miss in this case. The corresponding block is copied from lower level memory to cache line 3. Once the copy is done, the valid bit is turned to 'Y'and the tag is stored as 0. The corresponding information is returned back to the processor.

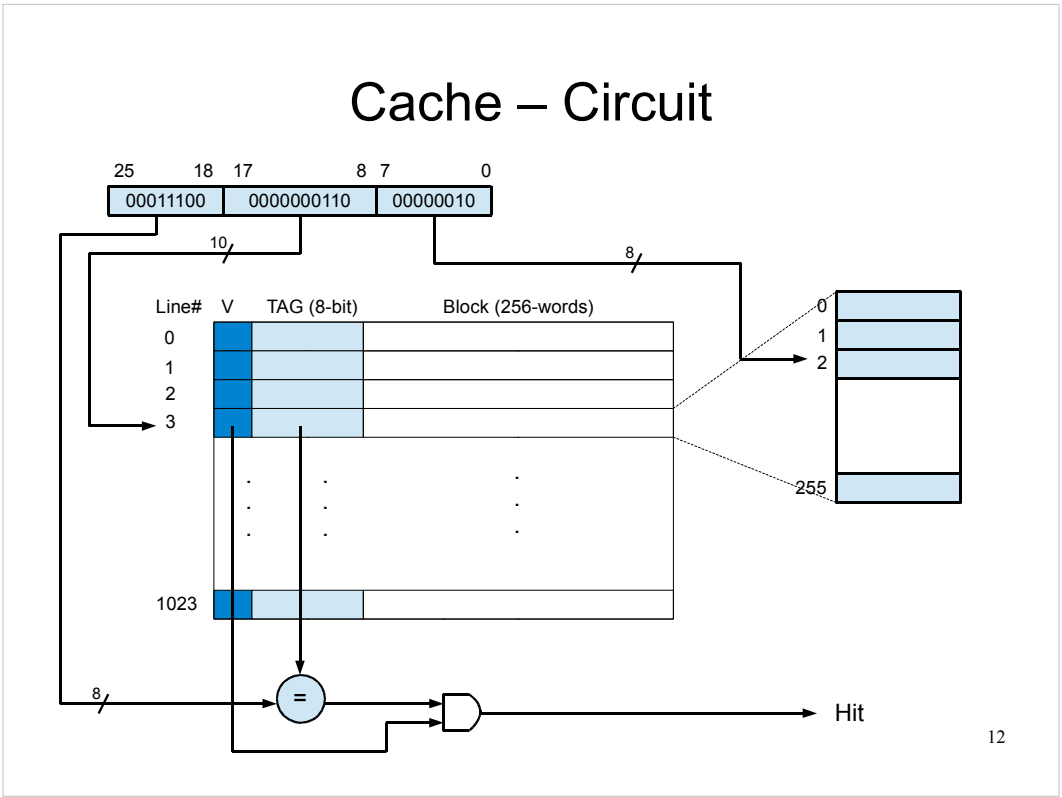
- For operation 7, cache line 0 is referenced which has a valid bit value 'Y' and tag as same at the required tag value 2. The the cache access is a hit and corresponding information is returned back to the processor.

- For operation 8, cache line 2 is referenced which is valid at the operation, but the stored tag 3 is different than required tag 2. The cache access is a miss in this case. The corresponding block is copied from lower level memory to cache line 2. Once the copy is done, the valid bit is turned to 'Y'and the tag is stored as 2. The corresponding information is returned back to the processor.

Cache Memory Circuit ...






- If number of cache line and the number of word in a block is integer power of 2 ($2^n$) we can easily avoid requirement of have quotient and remainder calculation circuit.

- Let's say there are 1K ($2^{10}$) cache line with block size 256 words ($2^8$). There are 64M ($2^{26}$) addressable words (just like in our projects) in the memory.

    - The information index within the 256-word line will be right most 8 bits of the given address. Rest of the bits (25:8) will represent the block address.
    - The cache line index will be number represented by (17:8).
    - Rest of the higher bits will represent tag (18:25)

- This also explains why (I,T) pair will be unique to an address. With such organization as represented here in this example, any block address (25:8) is nothing but concatenation of its tag T and cache index I ({I,T}). Since block addresses are unique from each other, the tag and index pair also should be unique for a given block.

- There is a equality comparator circuit to compare the stored tag and the computed tag from the requested address. If they are equal and the the valid bit is '1' then the 'Hit' signal goes high (implemented using the AND gate using the equality result bit and the valid bit). If it is hit, information is looked up the cache word line using the information index as in the lower bits of the address.

# CS147 - Lecture 20

Kaushik Patra
(kaushik.patra@sjsu.edu)

13

- Cache Structure
- Cache Operation
- Cache Circuit

*Reference Books / Source:*

1) *Chapter 4 of 'Computer Organization & Architecture' by Stallings*
2) *Chapter 7 of 'Computer Organization & Design' by Patterson and Hennessy/*