CS 222 Computer Organization & Architecture

Lecture 24 [19.03.2019]

Introduction to Cache Memory

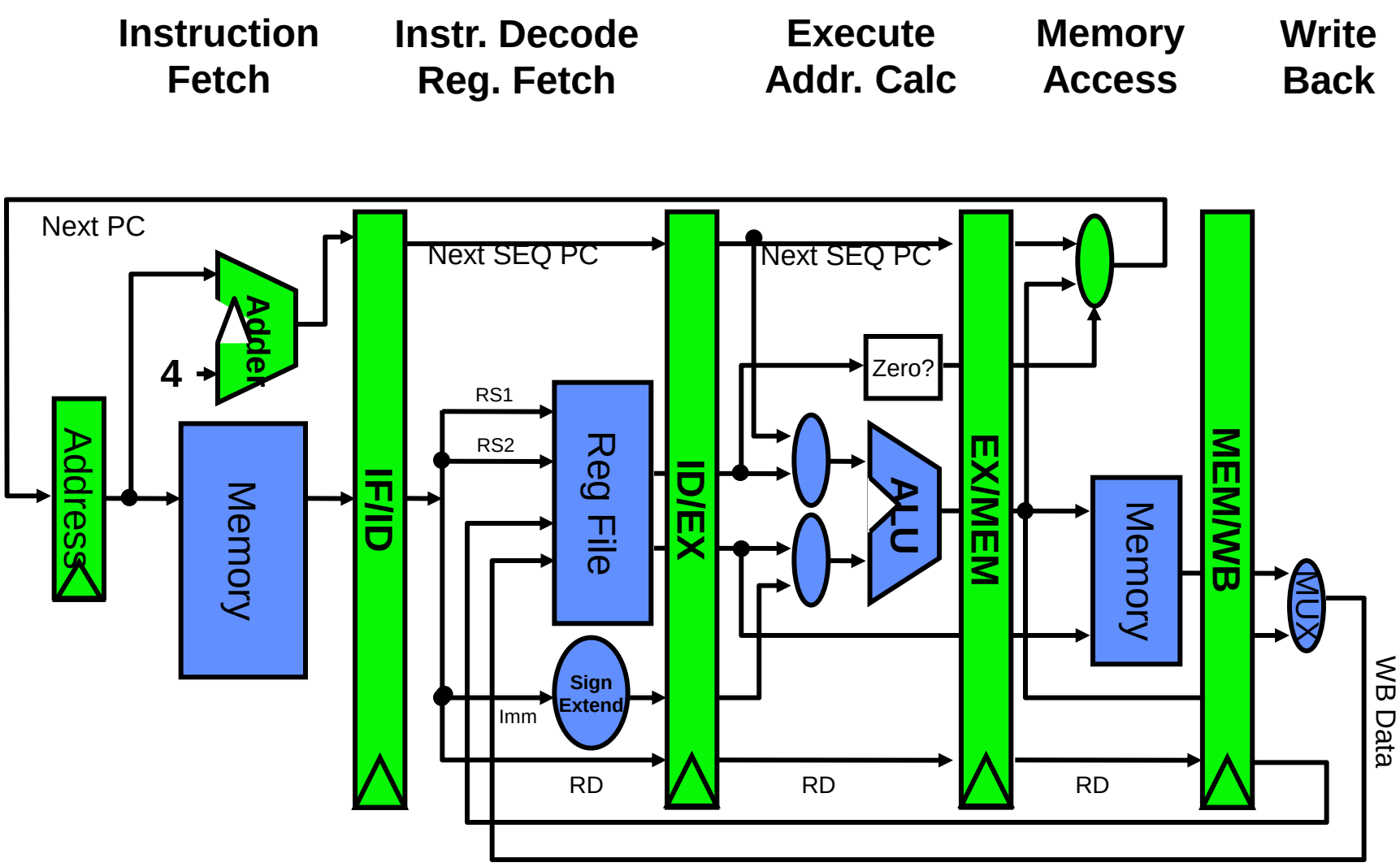
John Jose

Assistant Professor

Department of Computer Science & Engineering

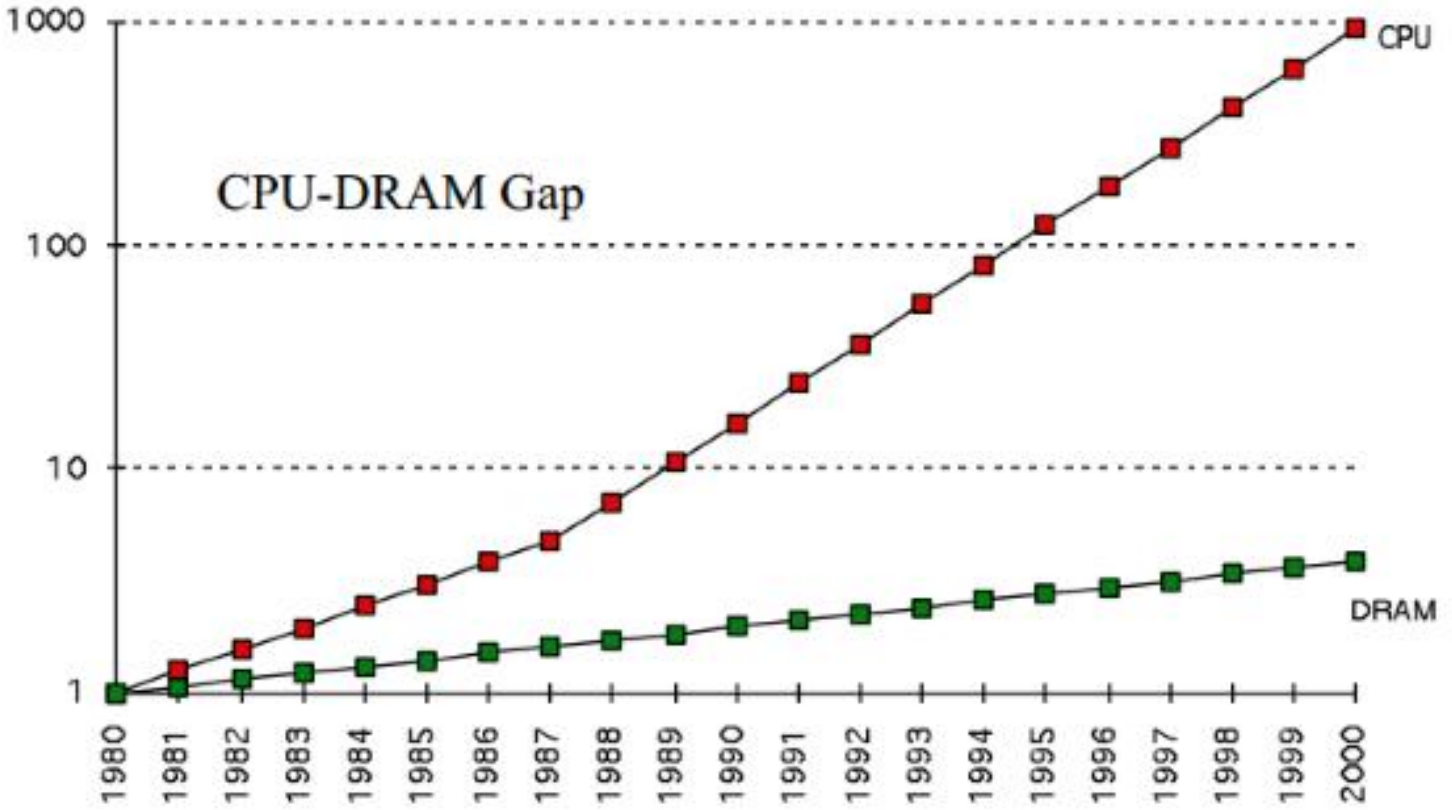
Indian Institute of Technology Guwahati, Assam.

Pipelined RISC Data path



Processor Memory Performance Gap

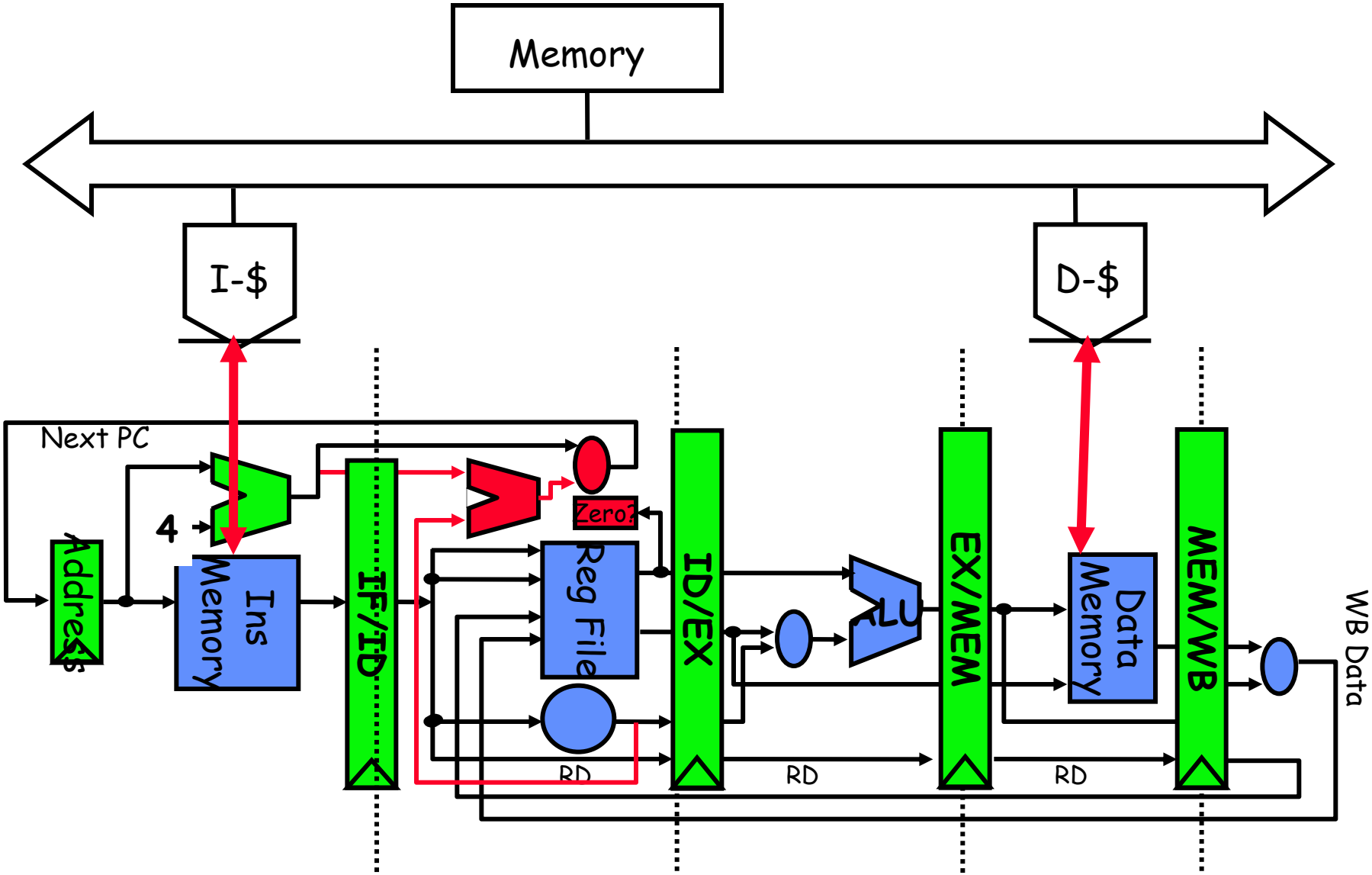
- Processor vs Memory Performance



1980: no cache in microprocessor;

1995 2-level cache

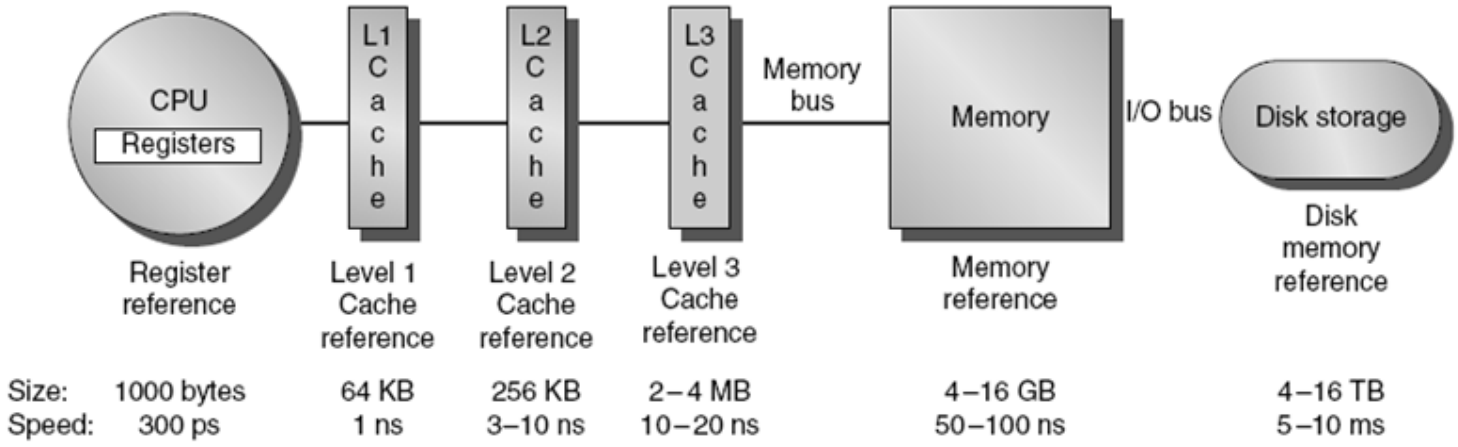
Relationship of Caches and Pipeline

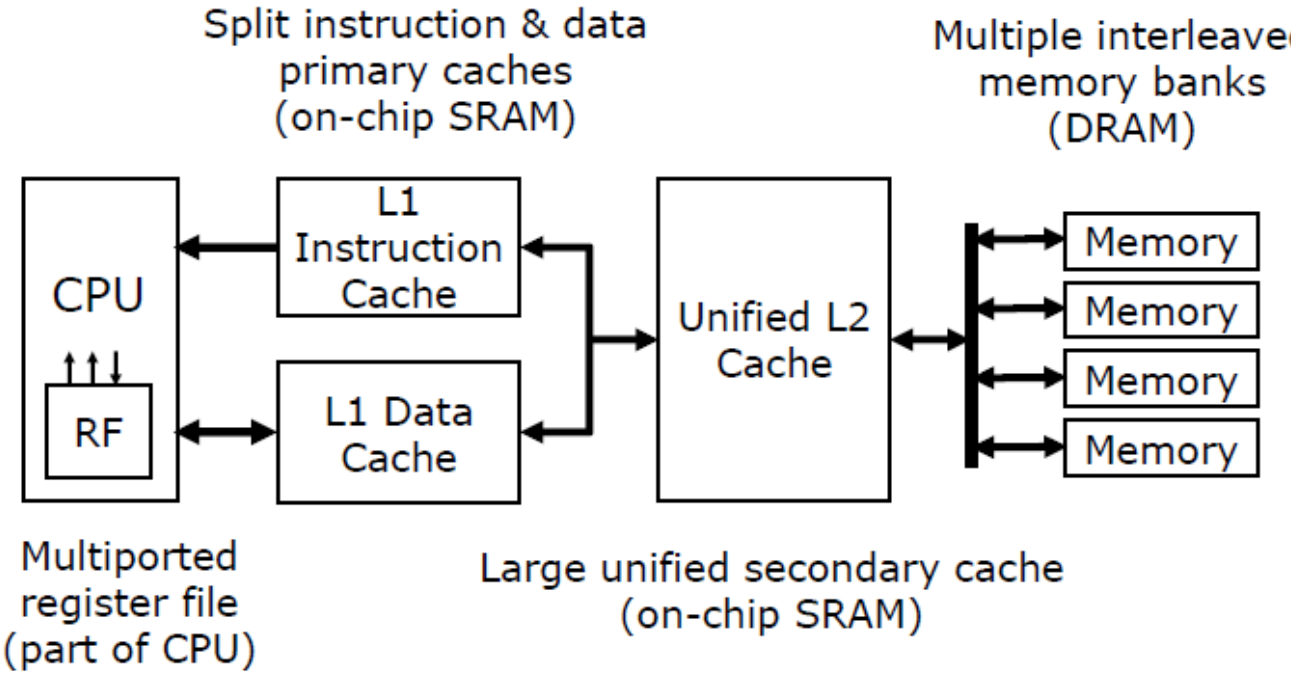


Role of memory

- Programmers always wanted unlimited amount of fast memory
- Create the illusion of a very large and fast memory
- Implement the memory of a computer as a hierarchy
- Multiple levels of memory with different speeds and sizes
- Entire addressable memory space available in largest, slowest memory
- Keep the smaller and faster memories close to the processor and the slower, large memory below that

Memory Hierarchy

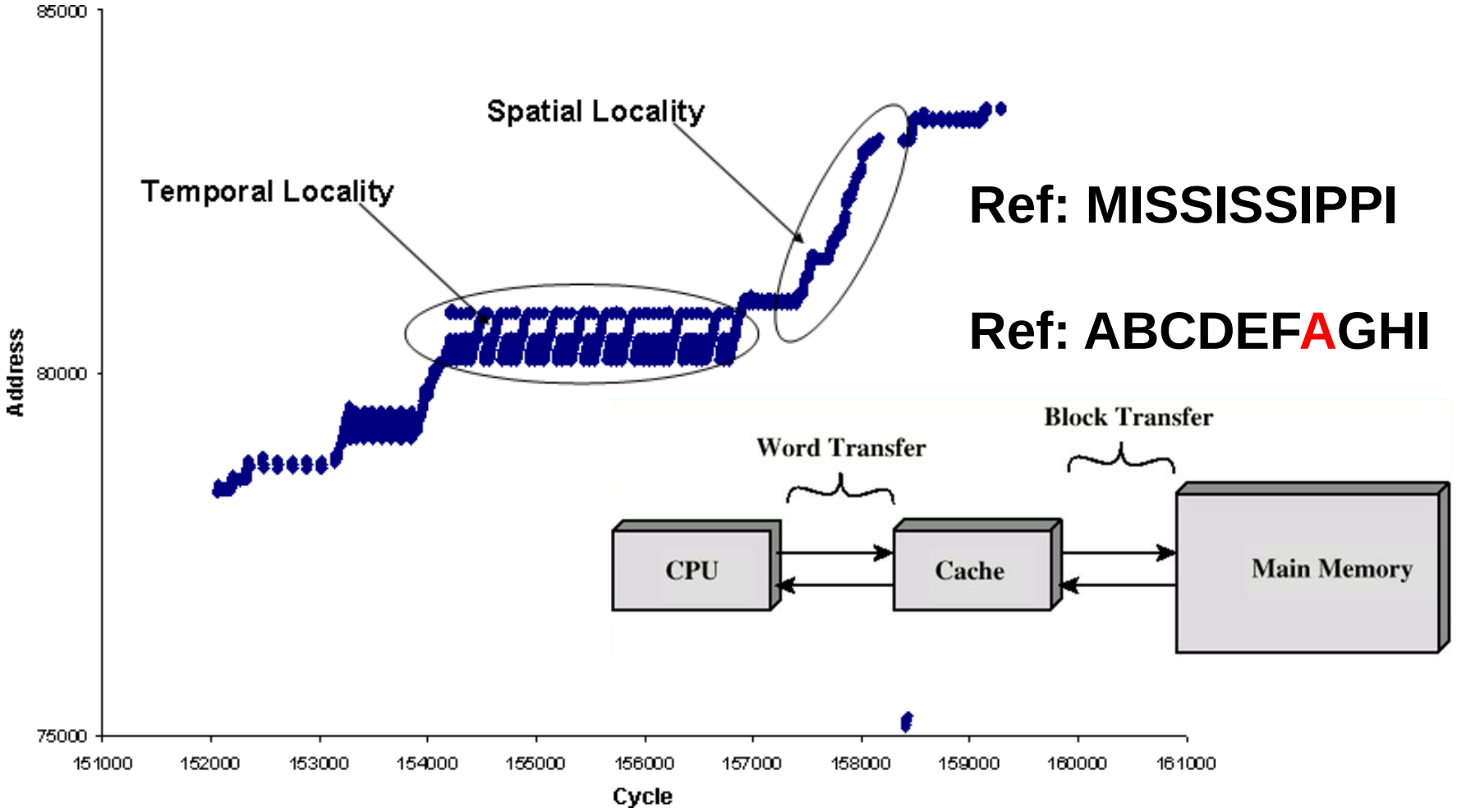




Cache Memory - Introduction

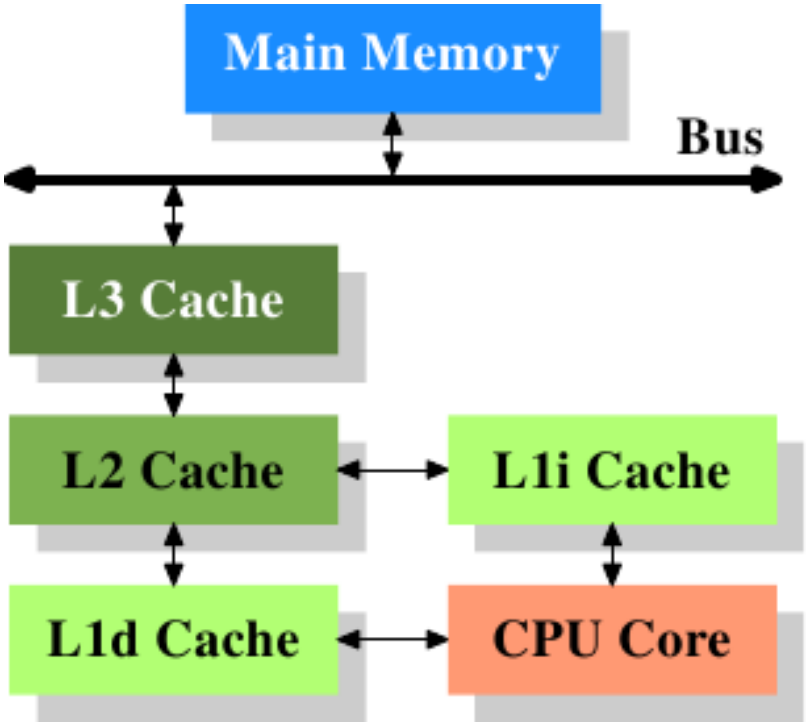
- Cache is a buffer between processor and memory
- Small but fast
- Old values will be removed from cache to make space for new values
- **Principle of Locality :** Programs access a relatively small portion of their address space at any instant of time
- **Temporal Locality :** If an item is referenced, it will tend to be referenced again soon
- **Spatial Locality :** If an item is referenced, items whose addresses are close by will tend to be referenced soon

Access Patterns



Cache Memory

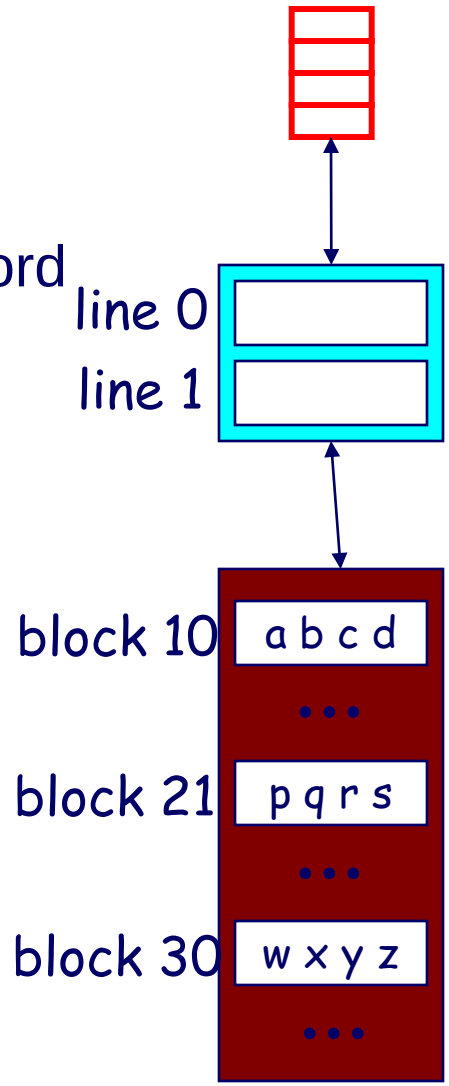
- Cache memories are small, fast SRAM-based memories managed in hardware by cache controller.
- It hold frequently accessed blocks of main memory
- CPU looks first for data in L1, then in L2, then in main memory.



CPU – Cache Interaction

The transfer unit between the CPU register file and the cache is a 4-byte word

The transfer unit between the cache and main memory is a 4-word block (16B)



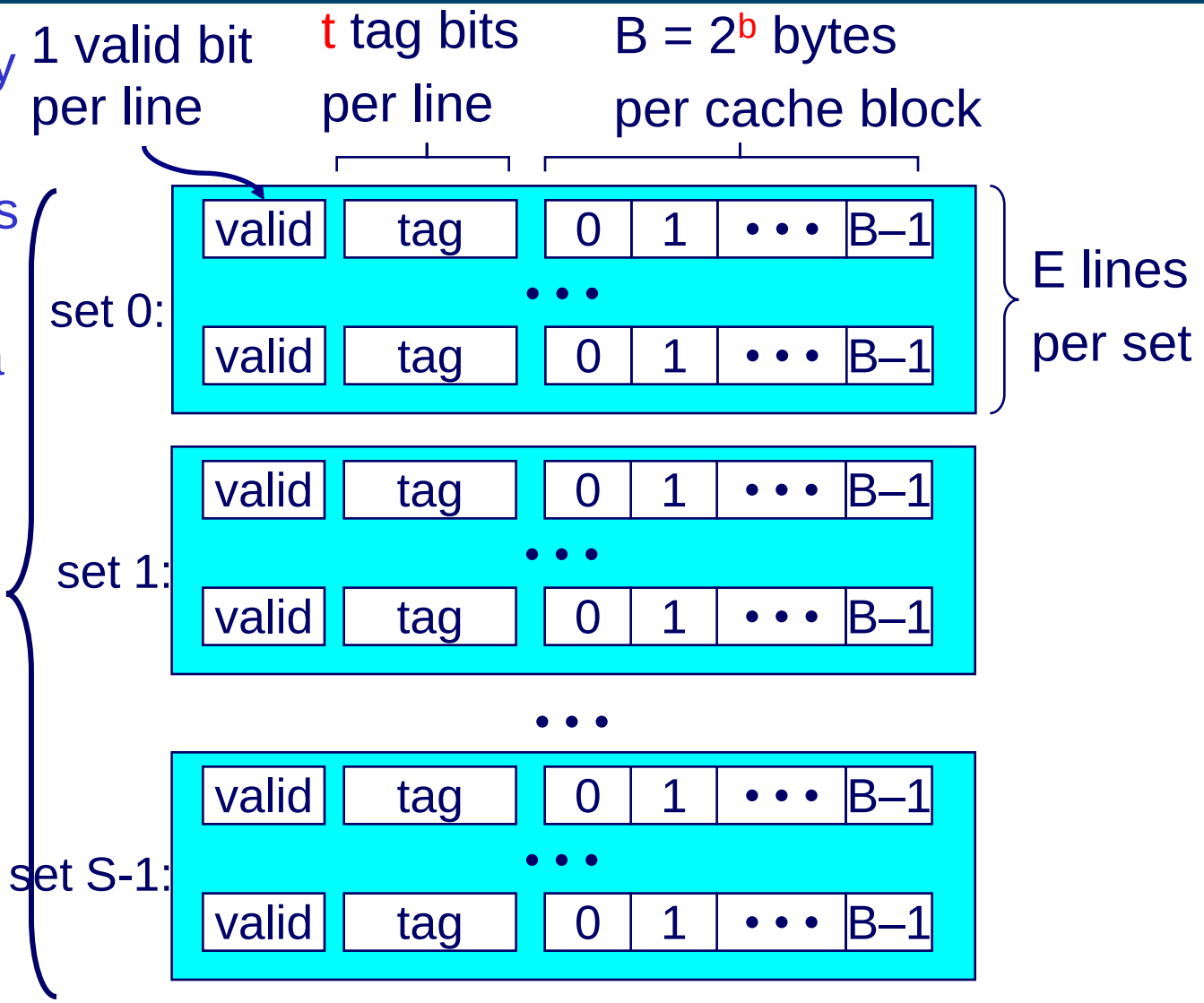
The tiny, very fast CPU register file has room for four 4-byte words

The small fast **L1 cache** has room for two 4-word blocks

The big slow main memory has room for many 4-word blocks

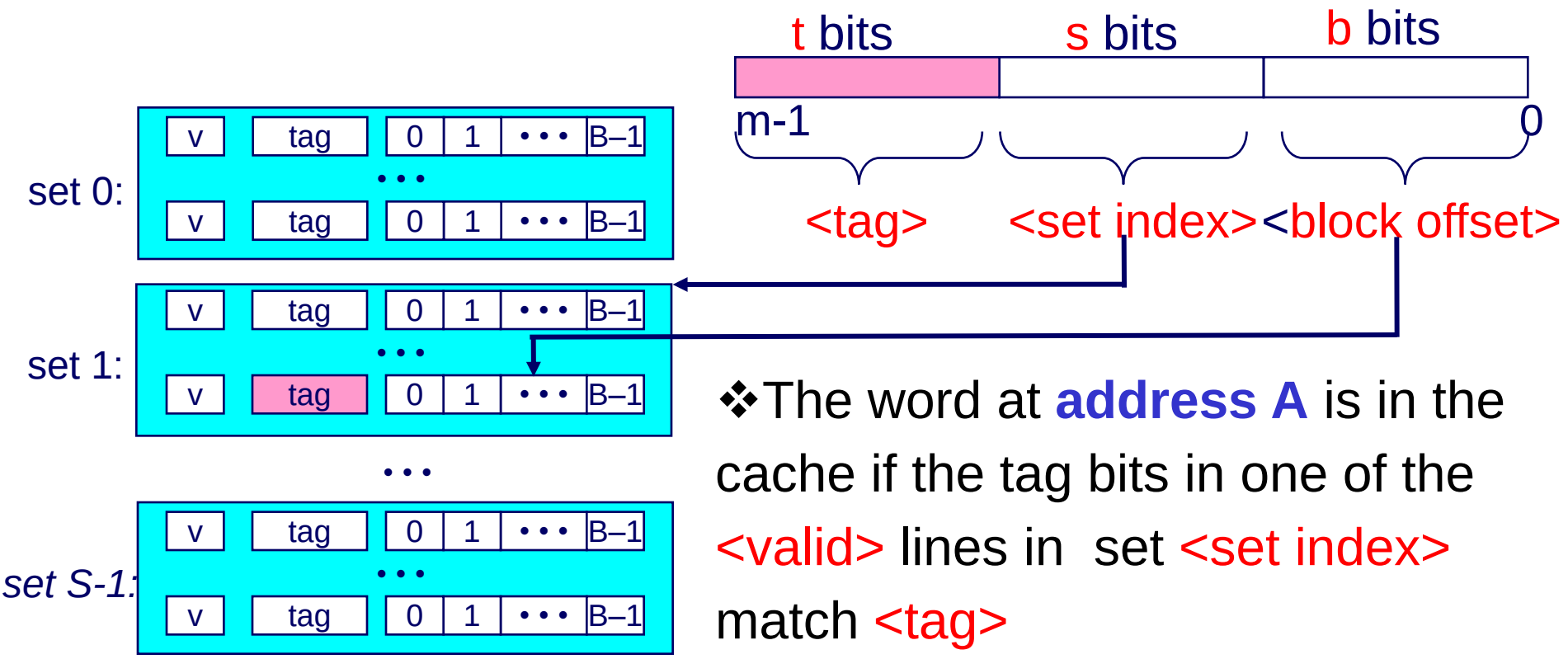
General Organization of a Cache

- Cache is an array of sets
- Each set contains one or more lines
- Each line holds a block of data



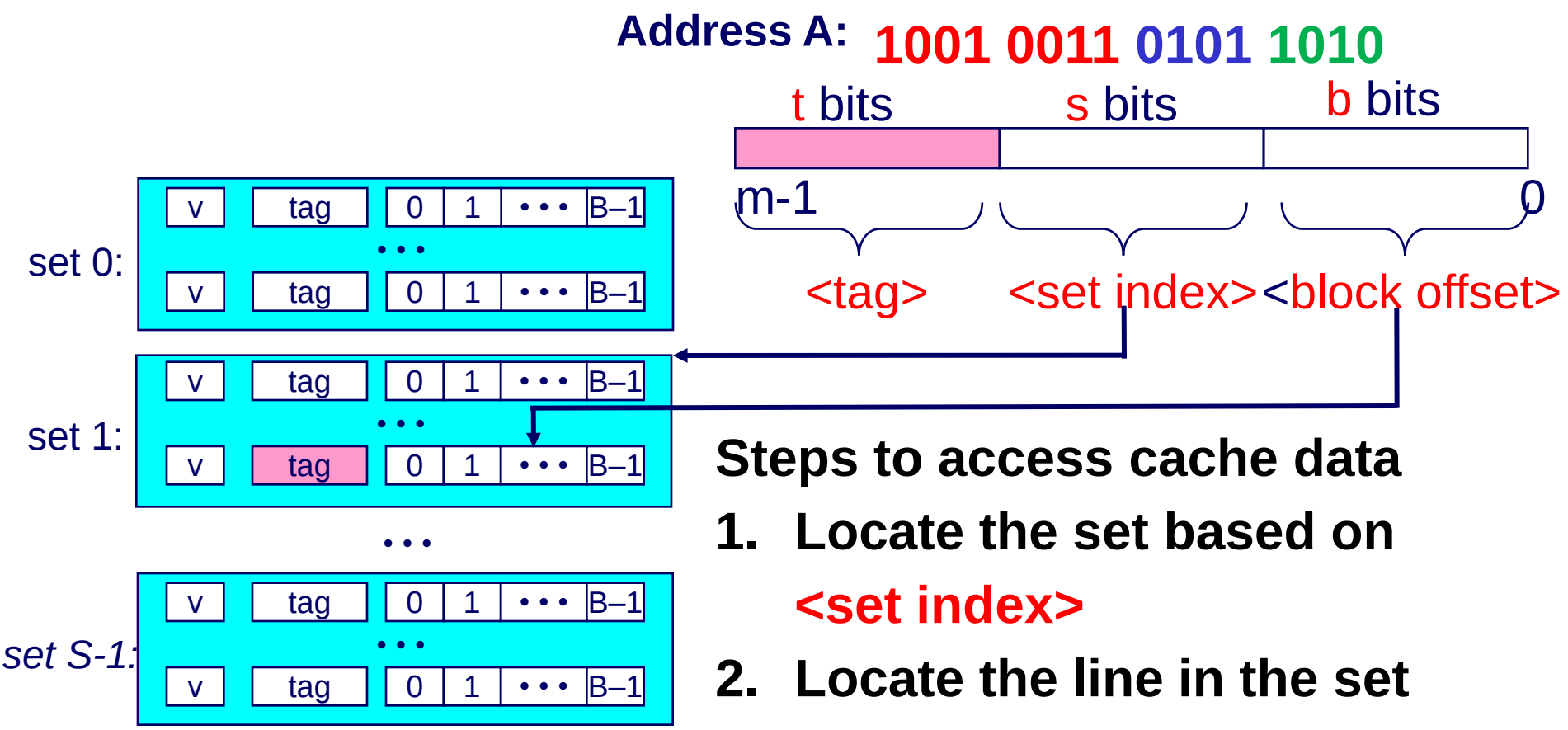
Addressing Caches

CPU wants → Address A:



- The word at **address A** is in the cache if the tag bits in one of the <valid> lines in set <set index> match <tag>
- The word contents begin at offset <block offset> bytes from the beginning of the block

Addressing Caches



Steps to access cache data

1. **Locate the set based on <set index>**
2. **Locate the line in the set based on <tag>**
3. **Check that the line is valid**
4. **Locate the data in the line based on <block offset>**

Basic Terminologies

- **Block :** Minimum unit of information that can be either present or not present in a cache level
- **Hit :** An access where the data requested by the processor is present in the cache
- **Miss :** An access where the data requested by the processor is not present in the cache
- **Hit Time :** Time to access the cache memory block and return the data to the processor.
- **Hit Rate / Miss Rate:** Fraction of memory access found (not found) in the cache
- **Miss Penalty :** Time to replace a block in the cache with the corresponding block from the next level.

Index and Offset Calculations

A cache has 512 KB capacity, 4B word, 64B block size and 8-way set associative. The system is using 32 bit address. Given the address 0X ABC89984, which set of cache will be searched and specify which word of the selected cache block will be forwarded if it is a hit in cache?

sets = CS/(BSxA) = $2^{19}/(2^6 x 2^3) = 2^{10}$ = 1024 sets
1 word = 4B , Hence 64 byte block has 16 words

Tag = 16	Index =10	Offset=6 (4+2)

0x ABC89984 = 1001 1001 1000 0100 → Set 614, word 1

0x 485669AC = 0110 1001 1010 1100 → Set 422, word 11

johnjose@iitg.ac.in
http://www.iitg.ac.in/johnjose/