# CS223 : Computer Architecture & Organization

## Lecture 19 [28.03.2022]

## Advanced Concepts in Cache Memory

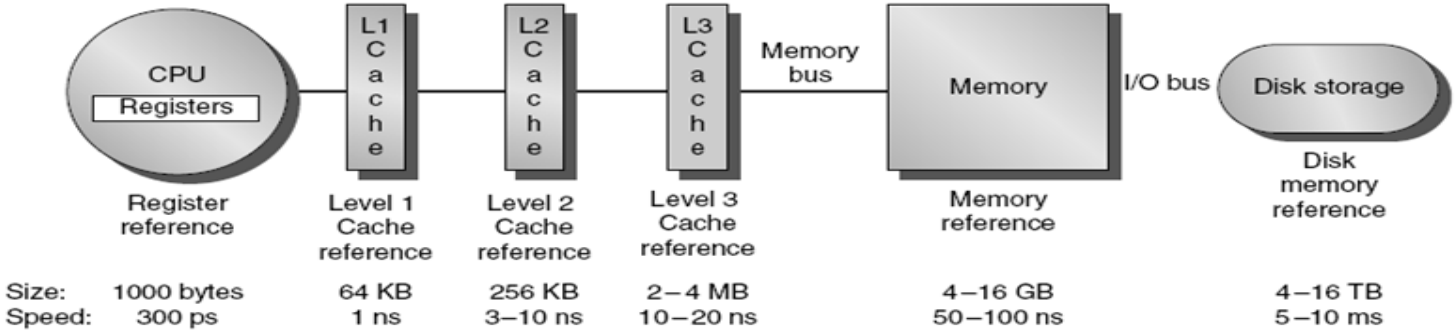

**Dr. John Jose**

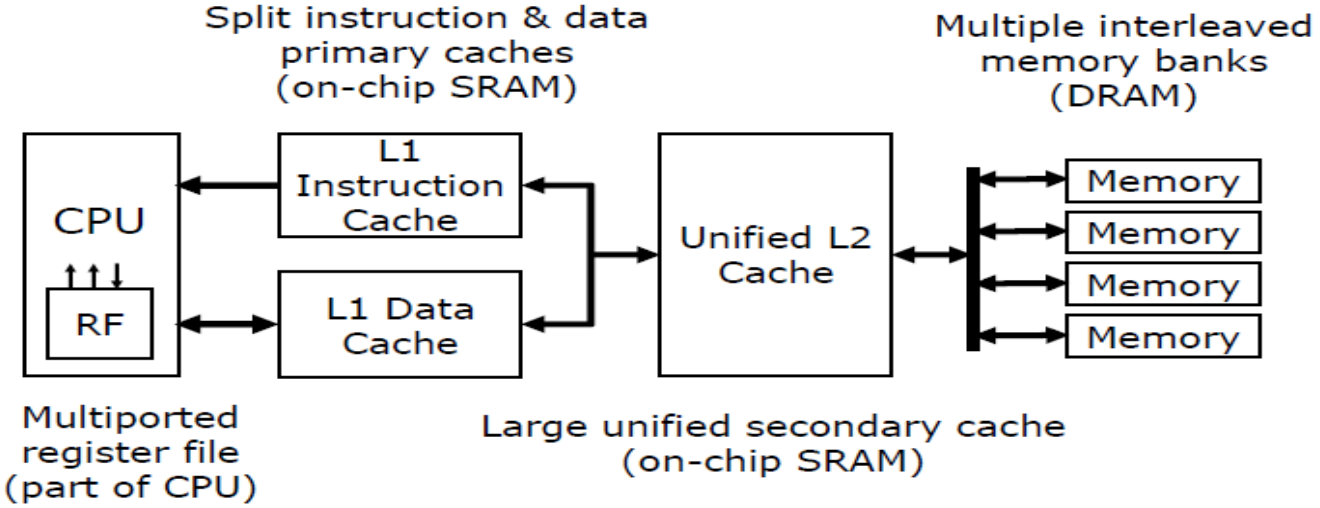
**Associate Professor**

**Department of Computer Science & Engineering**

**Indian Institute of Technology Guwahati, Assam.**

# Memory Hierarchy

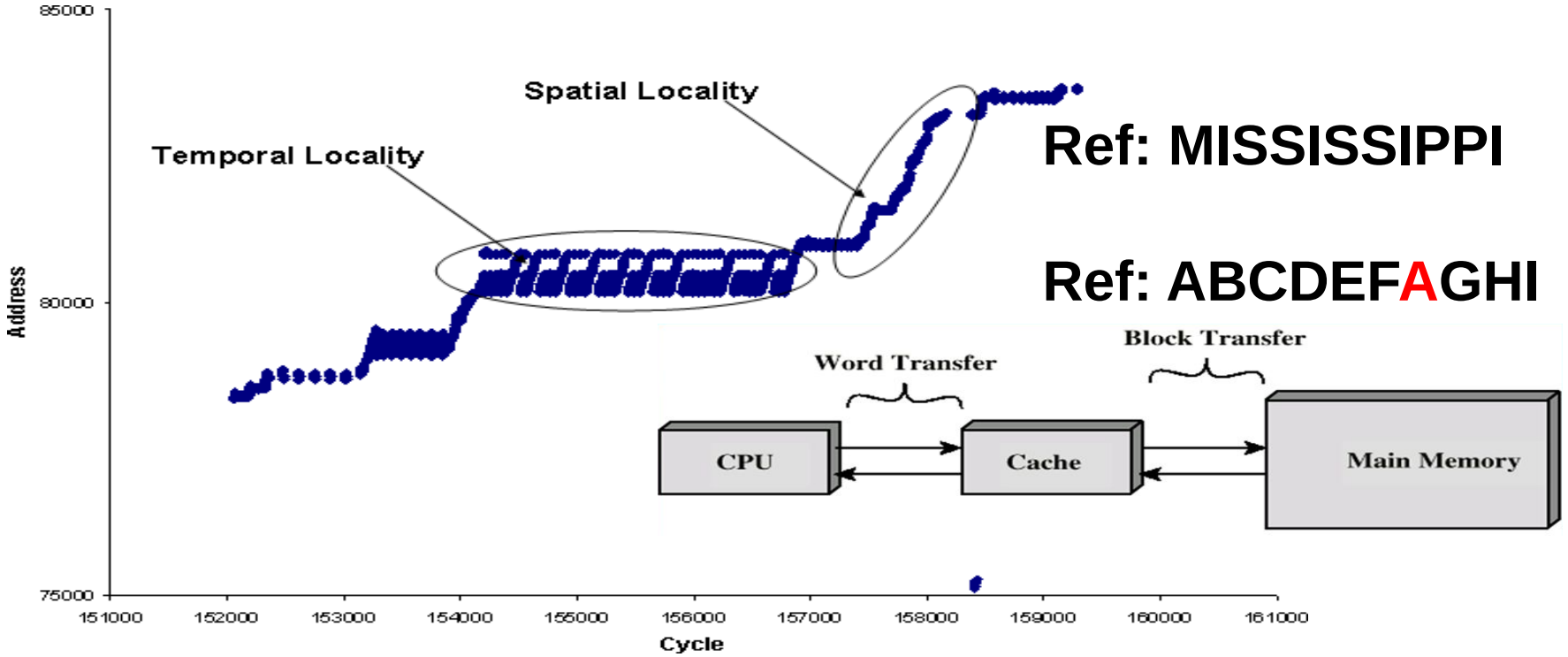


# Cache Memory - Introduction

- Cache is a small, fast buffer between processor and memory
- 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



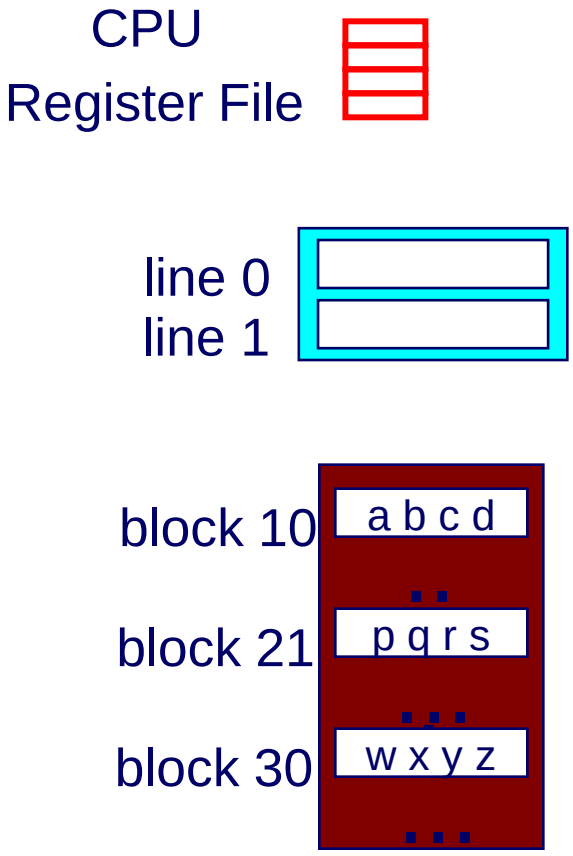

**Ref: MISSISSIPPI**

**Ref: ABCDEFAGHI**

# Cache Fundamentals

- **Block/Line :** 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.

# CPU – Cache Interaction

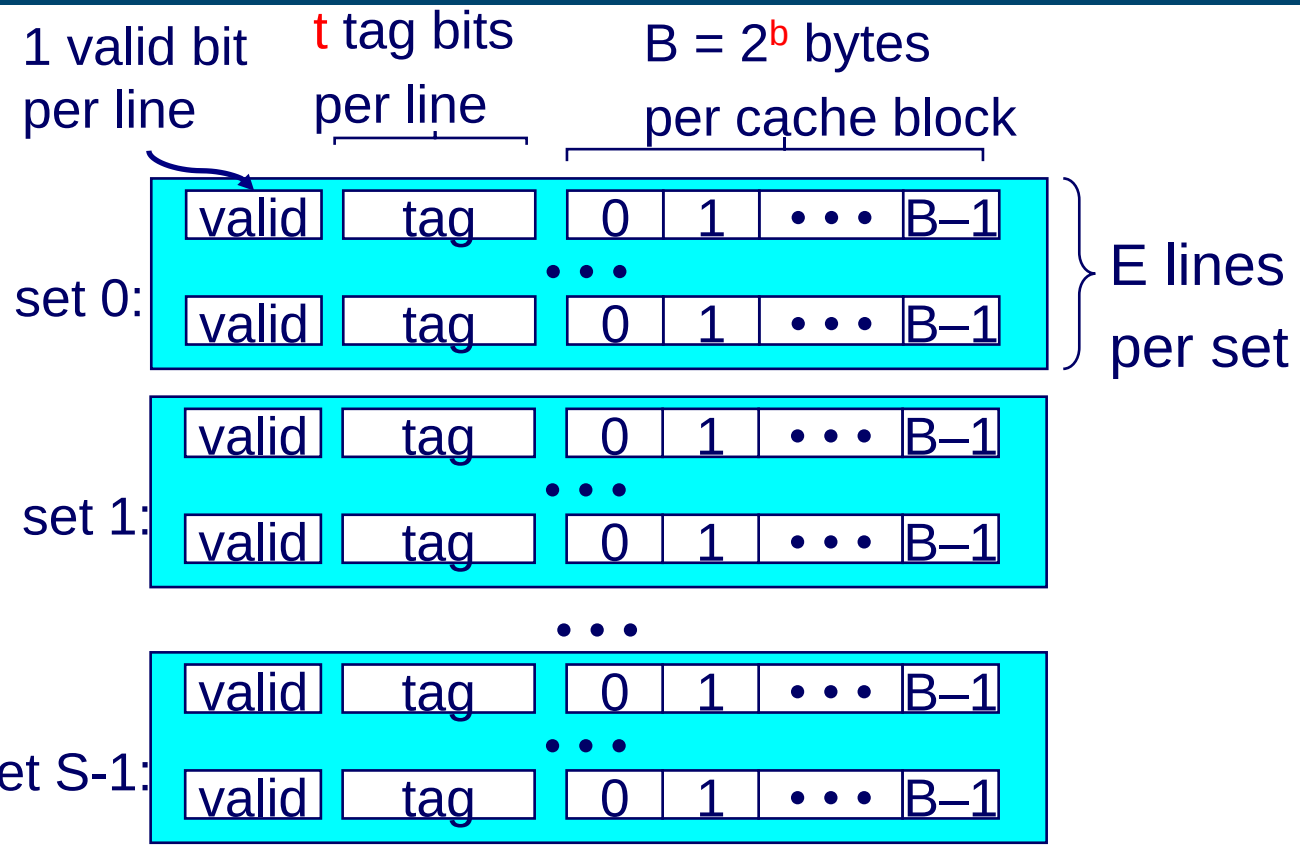



- Multiple very fast CPU registers
- The transfer unit between the CPU register file and the cache is a word
- The small fast **L1 cache** has room for multiple words/blocks
- The transfer unit between the cache and main memory is a block
- The big slow main memory has room for many blocks

# 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

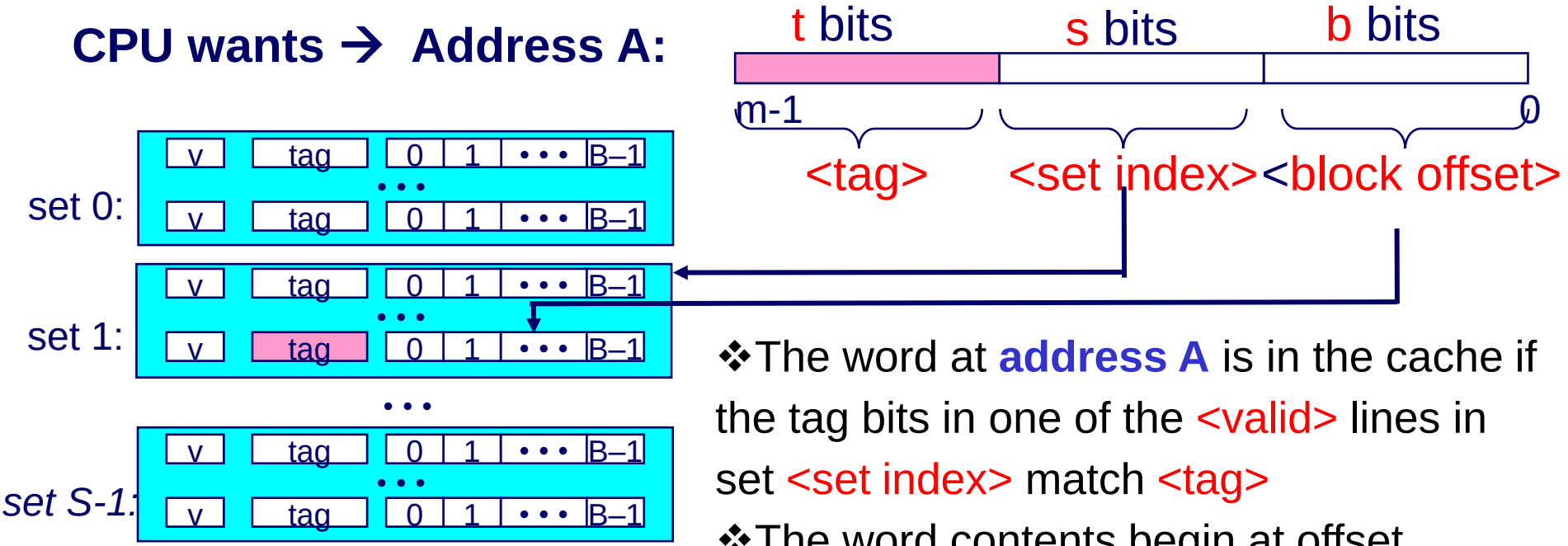
$S = 2^s$ sets




**Cache size: C = B x E x S data bytes**

# Addressing Caches

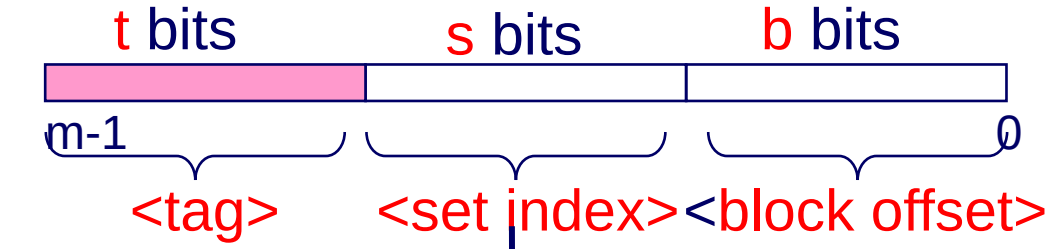
**CPU wants → Address A:**

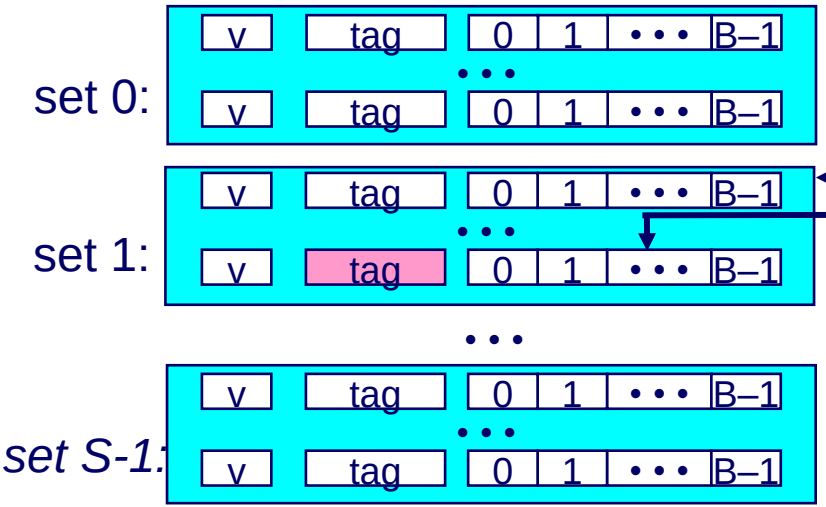



- The word at **address A** is in the cache if the tag bits in one of the <valid> lines in set <set index> match <tag>
- The word contents begin at offset <block offset> bytes from the beginning of the block

# Addressing Caches

**CPU wants → Address A:**



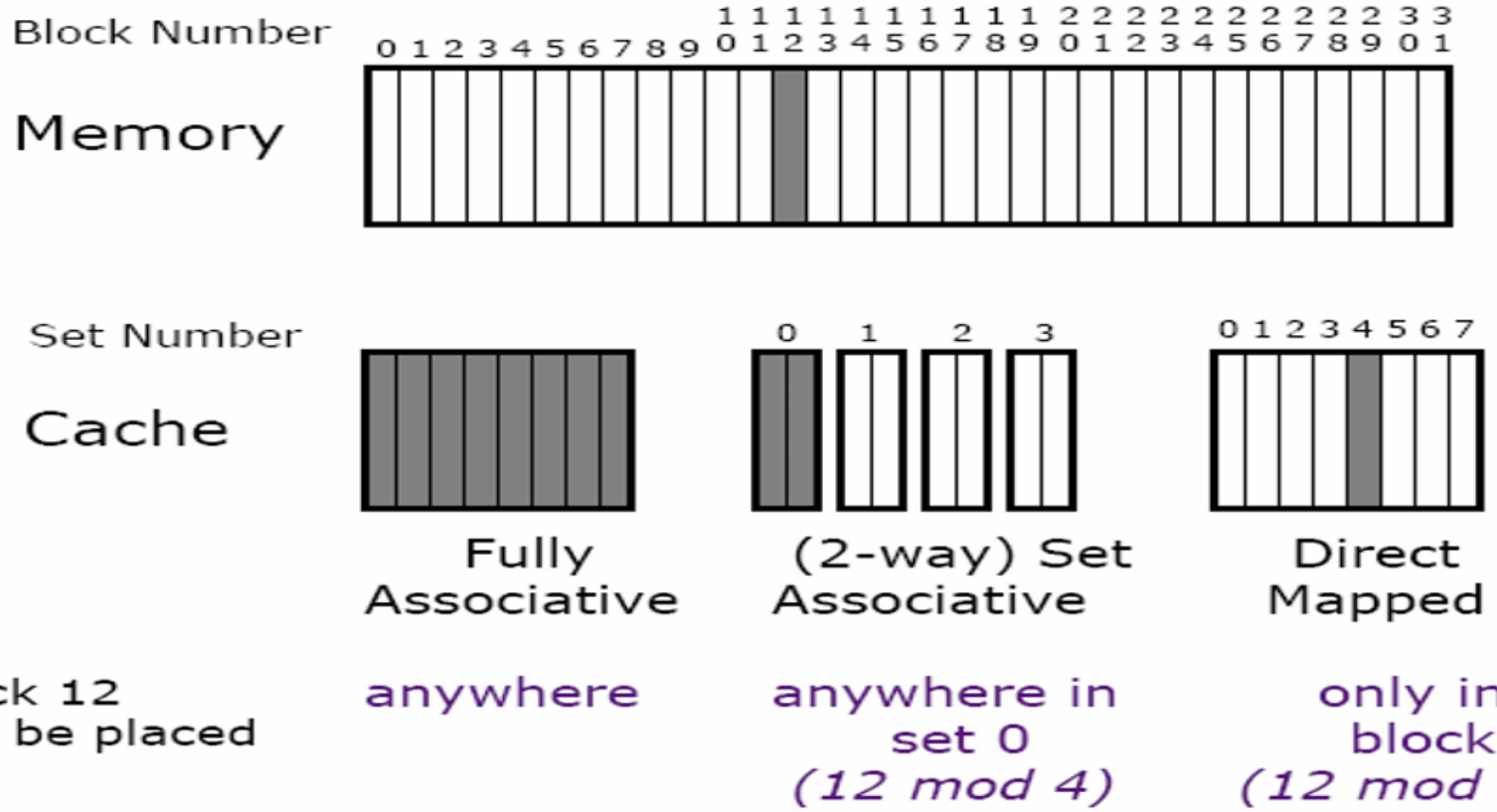




- Locate the set based on <set index>
- Locate the line in the set based on <tag>
- Check that the line is valid
- Locate the data in the line based on <block offset>

# Four cache memory design choices

- Where can a block be placed in the cache?
  - **Block Placement**
- How is a block found if it is in the upper level?
  - **Block Identification**
- Which block should be replaced on a miss?
  - **Block Replacement**
- What happens on a write?
  - **Write Strategy**

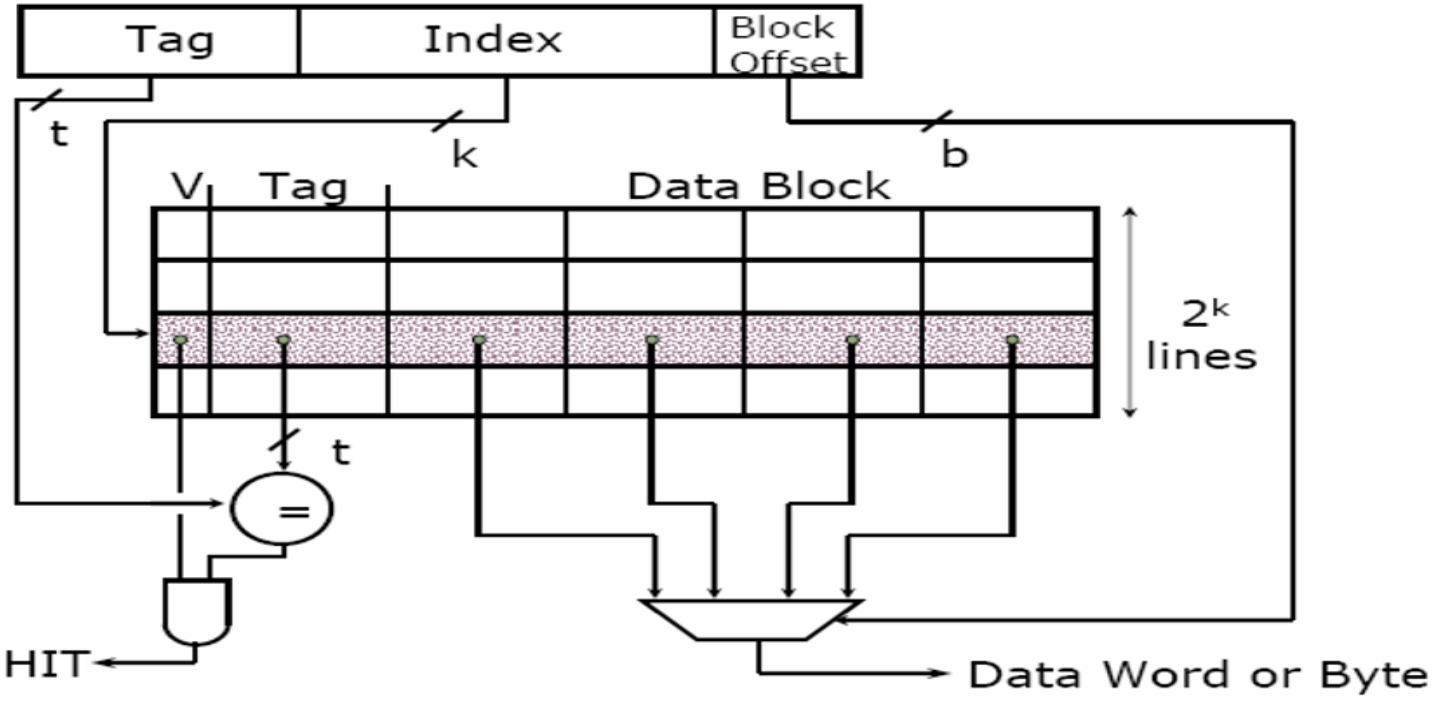
# Block Placement



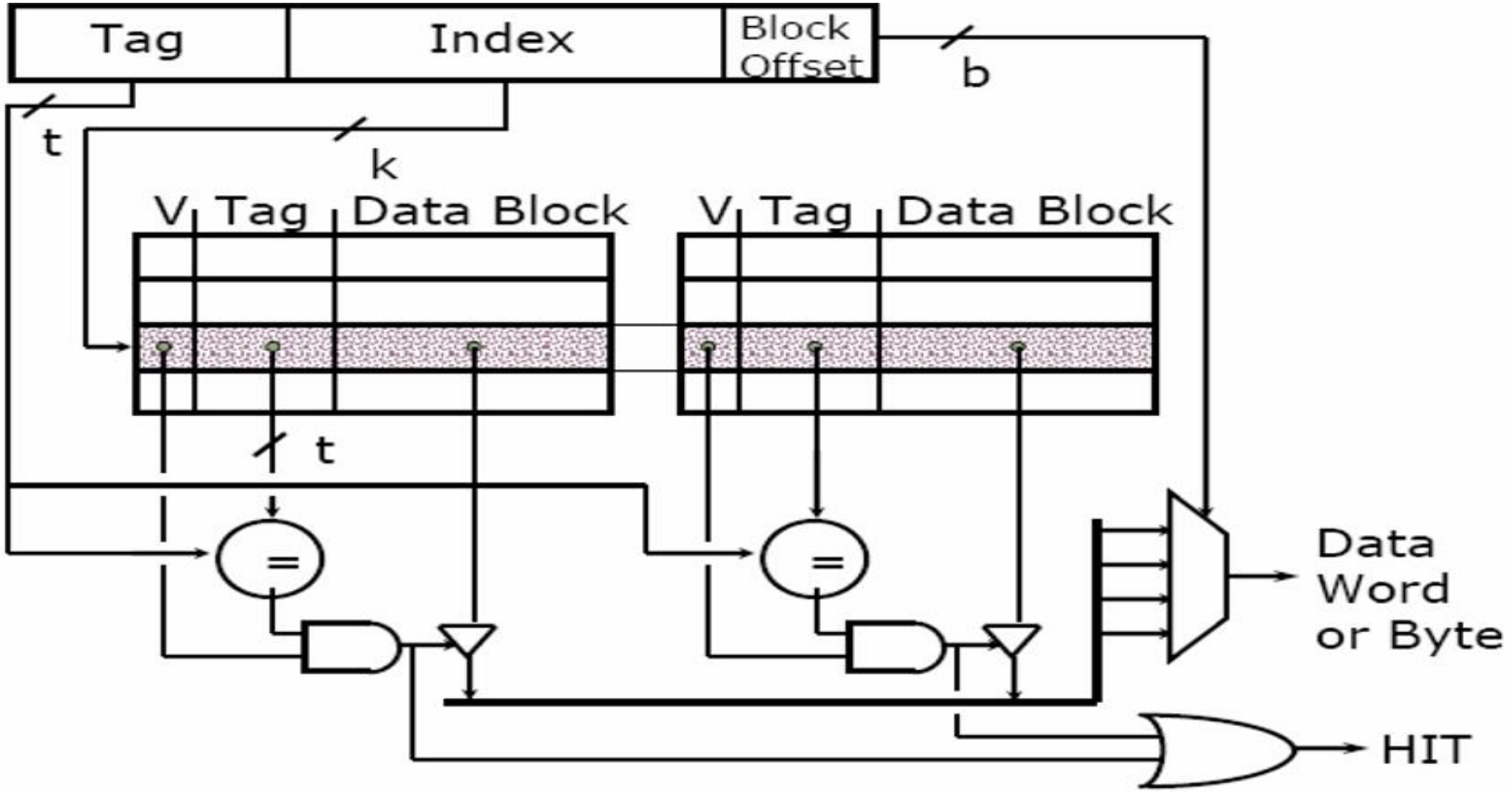
# Cache Mapping / Block Placement

- **Direct mapped**
  - Block can be placed in only one location
  - (Block Number) **Modulo** (Number of blocks in cache)
- **Set associative**
  - Block can be placed in one among a list of locations
  - (Block Number) **Modulo** (Number of sets)
- **Fully associative**
  - Block can be placed anywhere

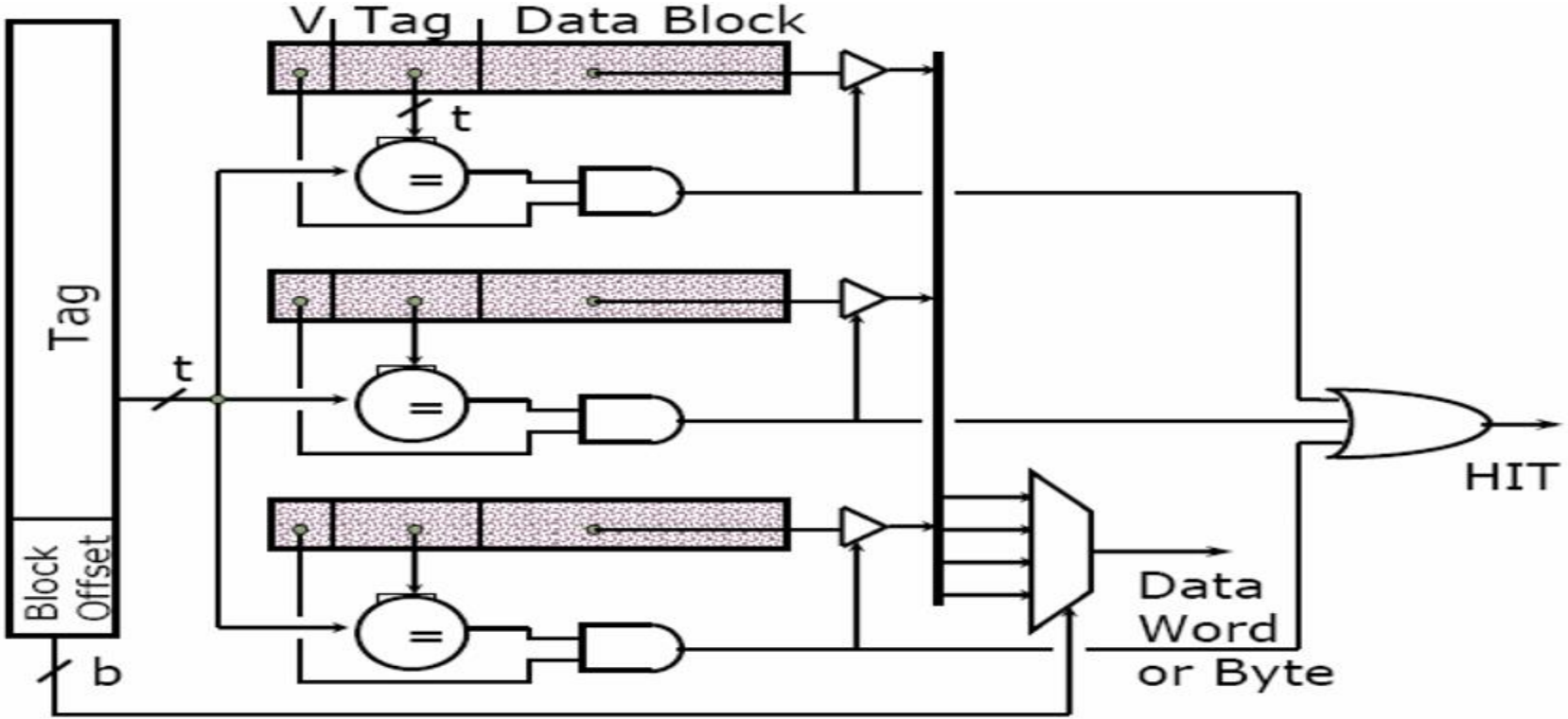
# Block Identification – Direct mapped



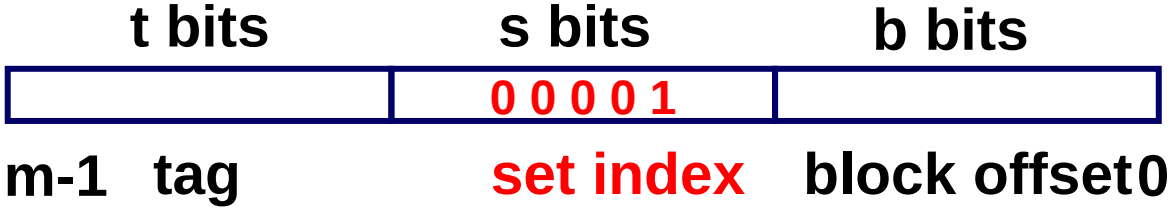
# Block Identification – Set Associative



# Block Identification – Fully Associative

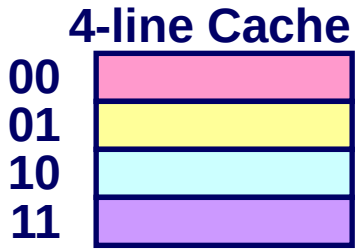


# Cache Indexing



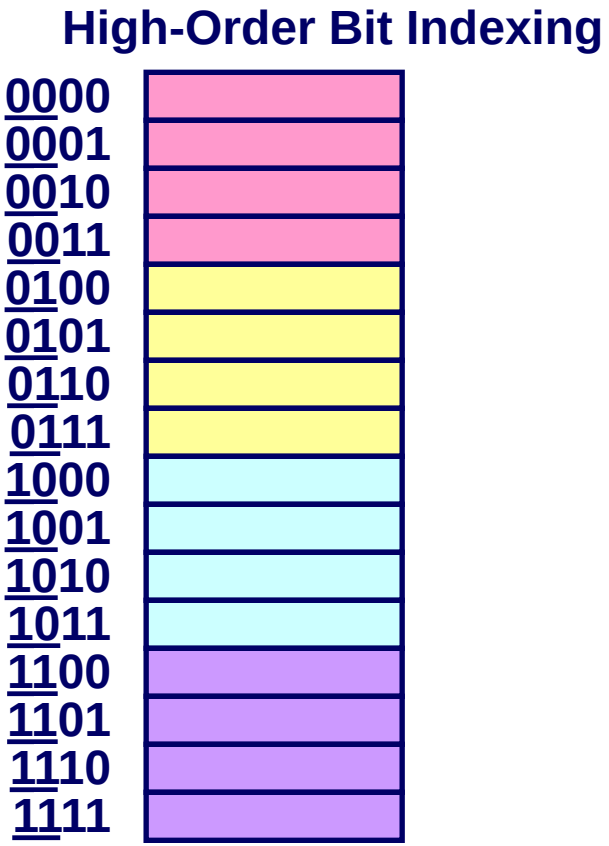

- Decoders are used for indexing
- Indexing time depends on decoder size ( s: $2^s$)
- Smaller number of sets, less indexing time.

# Why Use Middle Bits as Index?

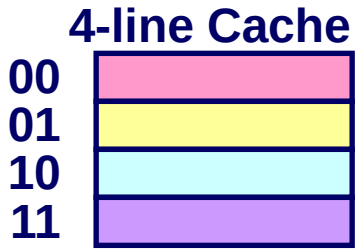



## High-Order Bit Indexing

- Adjacent memory lines would map to same cache entry
- Poor use of spatial locality

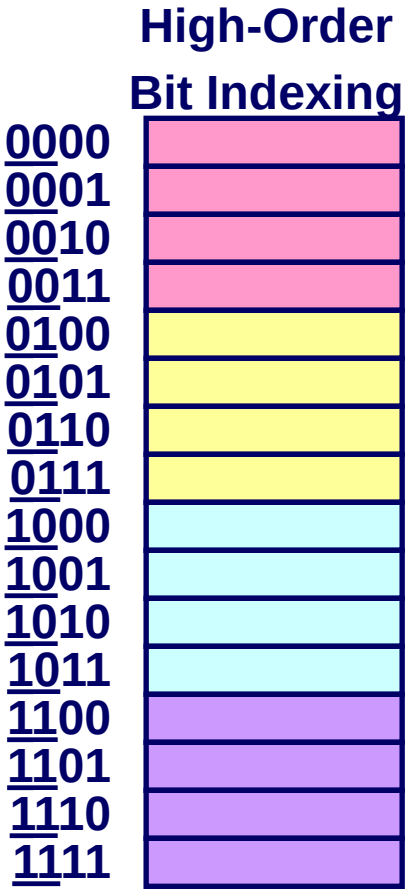


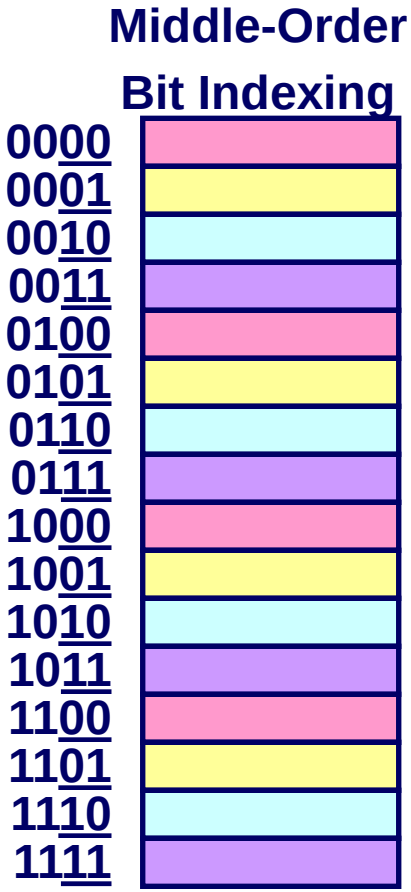
# Why Use Middle Bits as Index?




## Middle-Order Bit Indexing

- Consecutive memory lines map to different cache lines
- Better use of spacial locality without replacement



# Reference

- **Computer Architecture-A Quantitative Approach** (5th edition), John L. Hennessy, David A. Patterson, Morgan Kaufman.
- Chapter 2: Memory Hierarchy Design
  - Section 2.1: Introduction
- Appendix B: Review of Memory Hierarchy
  - Section B.1: Introduction
- **NPTEL Video Links:**
  - **https://tinyurl.com/yet9tlmo**
  - **https://tinyurl.com/yjx6yx4m**


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