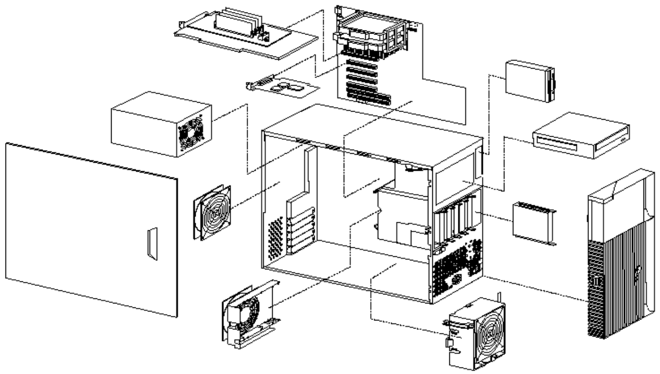
EDU
EAST DELTA
UNIVERSITY

Department of Computer Science and Engineering
School of Science, Engineering & Technology



CSE 317: Computer Organization & Architecture

Wahidul Alam, Lecturer, CSE, SoSET, EDU

Topic 6 – Cache Memory

- Key Characteristics of Computer Memory Systems
- Cache and Main Memory
- Elements of Cache Memory

Key Characteristics of Computer Memory Systems

- Location
- Capacity
- Unit of transfer
- Access method
- Performance
- Physical type
- Physical characteristics
- Organization

Location

- Internal (e.g. processor registers, cache. main memory)
- External (e.g. optical disks, magnetic disks, tapes)

Capacity

- Word size

—The natural unit of organization

- Number of words

—or Bytes

Unit of Transfer

- Internal

—Usually governed by data bus width

- External

—Usually a block which is much larger than a word

- Addressable unit

—Smallest location which can be uniquely addressed

—Word internally

—Cluster on M$ disks

Access Methods (1)

- Sequential

—Start at the beginning and read through in order

—Access time depends on location of data and previous location

—e.g. tape

- Direct

—Individual blocks have unique address

—Access is by jumping to vicinity plus sequential search

—Access time depends on location and previous location

—e.g. disk

Access Methods (2)

- Random

—Individual addresses identify locations exactly

—Access time is independent of location or previous access

—e.g. RAM

- Associative

—Data is located by a comparison with contents of a portion of the store

—Access time is independent of location or previous access

—e.g. cache

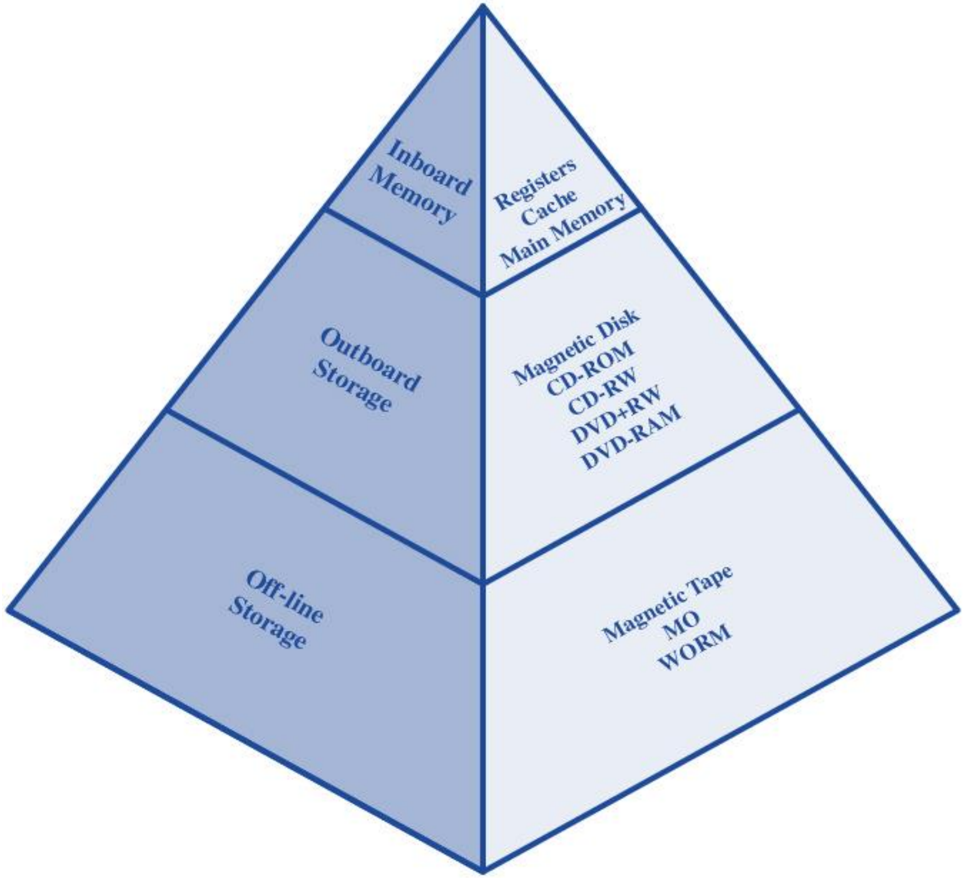
Memory Hierarchy

- Registers
  - In CPU

- Internal or Main memory
  - May include one or more levels of cache
  - “RAM”

- External memory
  - Backing store

Memory Hierarchy - Diagram



Performance

- Access time

—Time between presenting the address and getting the valid data

- Memory Cycle time

—Time may be required for the memory to “recover” before next access

—Cycle time is access + recovery

- Transfer Rate

—Rate at which data can be moved

Physical Types

- Semiconductor

—RAM

- Magnetic

—Disk & Tape

- Optical

—CD & DVD

- Others

—Bubble

—Hologram

Physical Characteristics

- Decay
- Volatility
- Erasable
- Power consumption

The Bottom Line

- How much?

—Capacity

- How fast?

—Time is money

- How expensive?

Hierarchy List

- Registers
- L1 Cache
- L2 Cache
- Main memory
- Disk cache
- Disk
- Optical
- Tape

So you want fast?

- It is possible to build a computer which uses only static RAM (see later)

- This would be very fast

- This would need no cache

—How can you cache cache?

- This would cost a very large amount

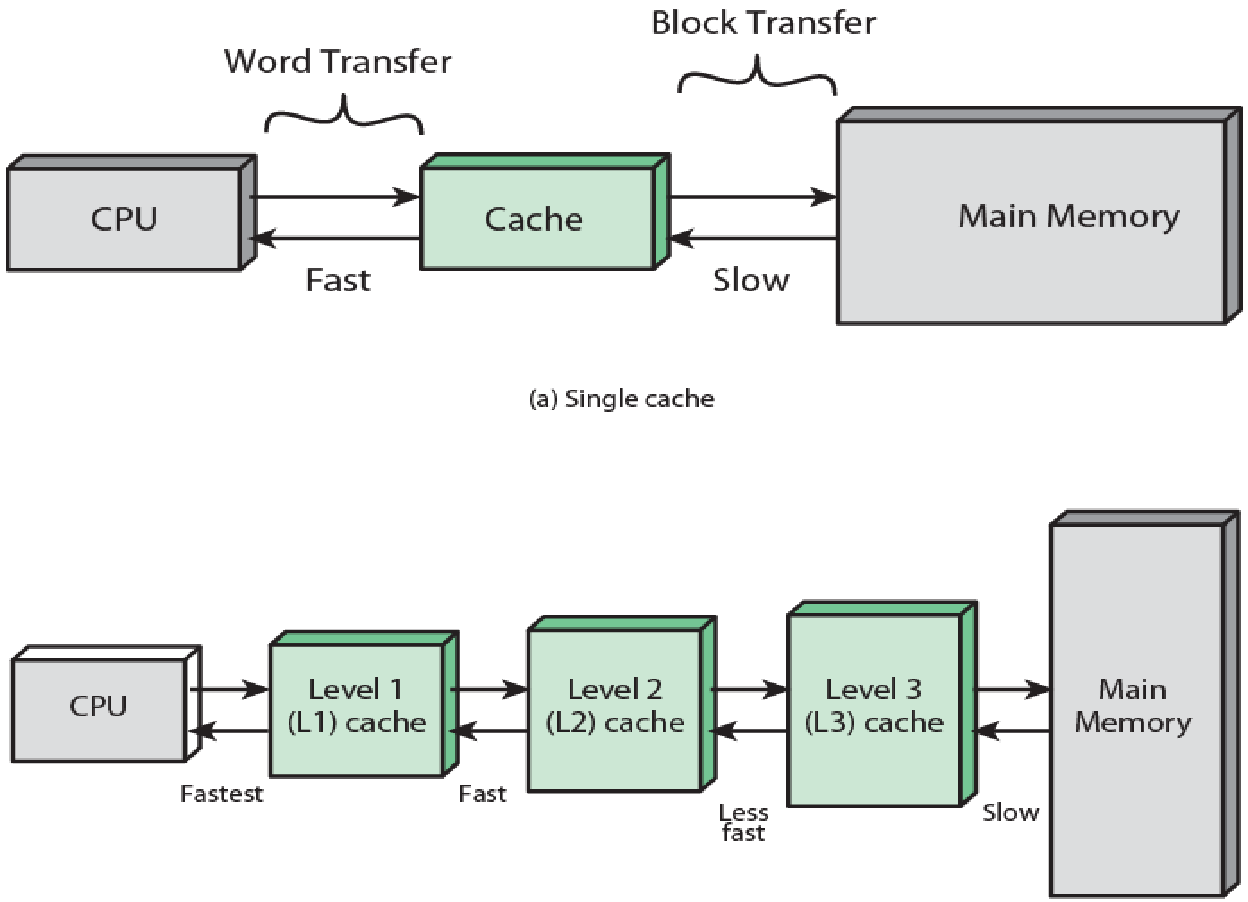
Locality of Reference

- During the course of the execution of a program, memory references tend to cluster
- e.g. loops

Cache

- Small amount of fast memory
- Sits between normal main memory and CPU
- May be located on CPU chip or module

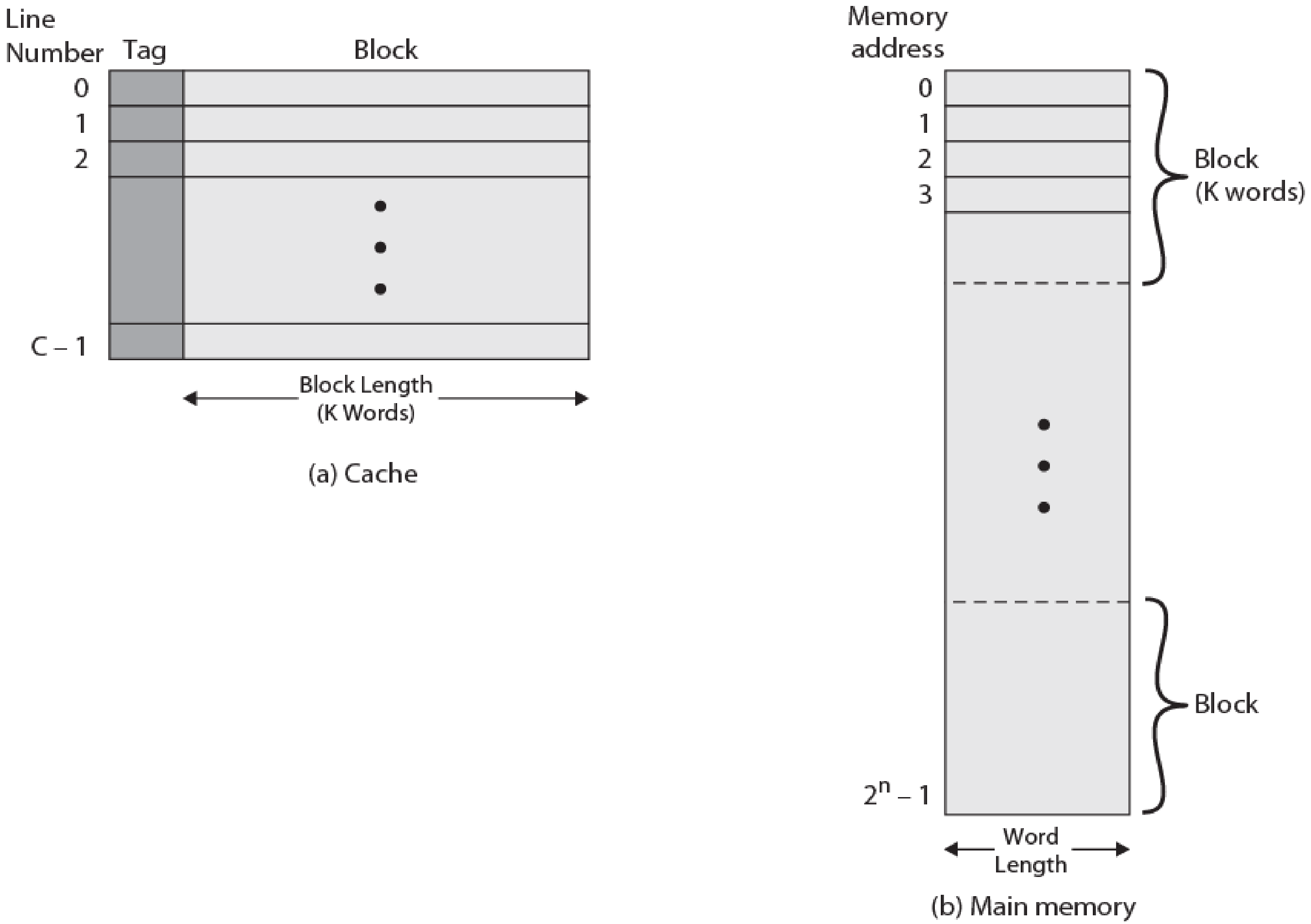
Cache and Main Memory



(a) Single cache

(b) Three-level cache organization

Cache/Main Memory Structure



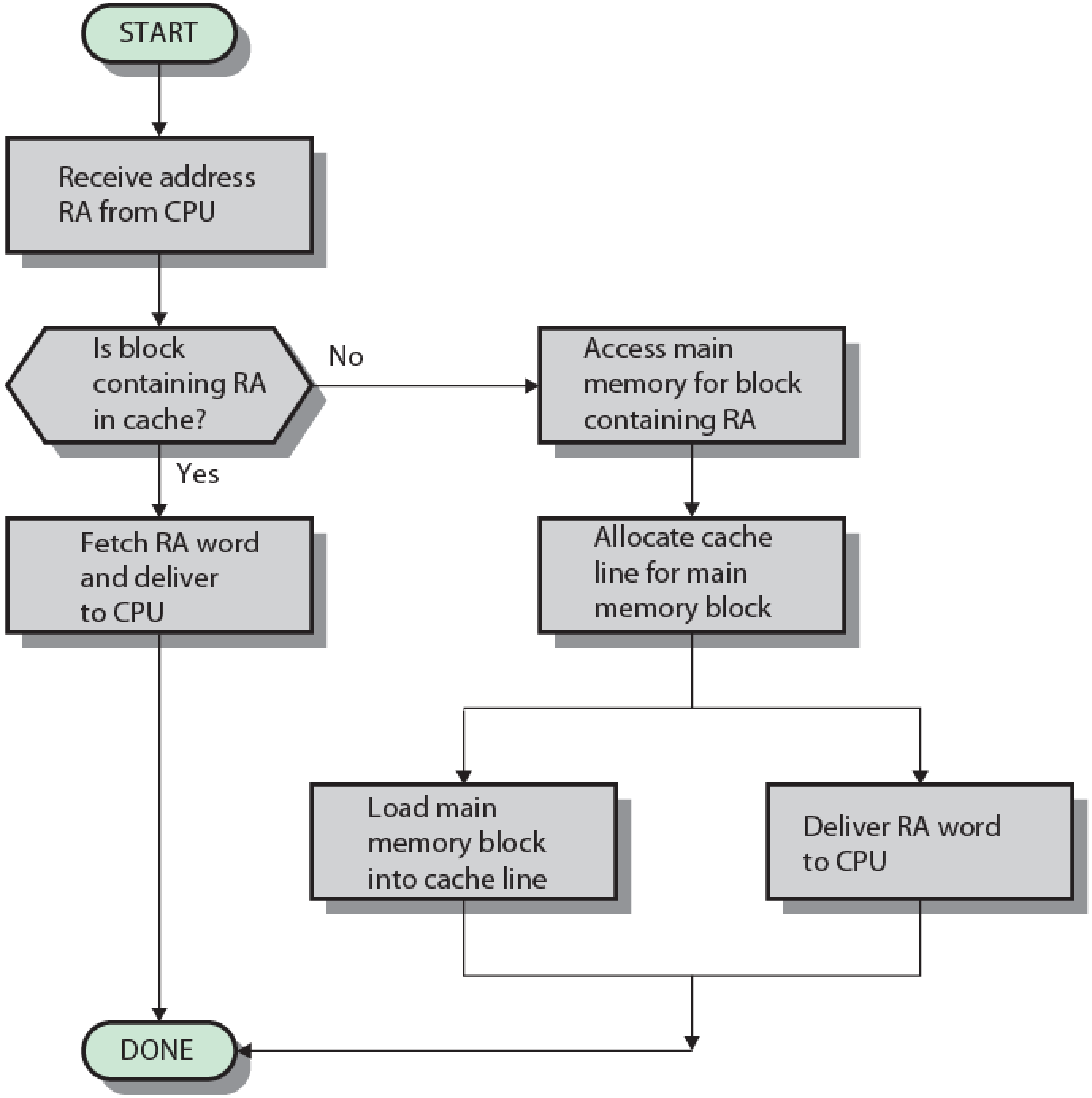
(a) Cache

(b) Main memory

Cache operation – overview

- CPU requests contents of memory location
- Check cache for this data
- If present, get from cache (fast)
- If not present, read required block from main memory to cache
- Then deliver from cache to CPU
- Cache includes tags to identify which block of main memory is in each cache slot

Cache Read Operation - Flowchart


START
Receive address RA from CPU
Is block containing RA in cache?
No
Access main memory for block containing RA
Yes
Fetch RA word and deliver to CPU
Allocate cache line for main memory block
Load main memory block into cache line
Deliver RA word to CPU
DONE

Cache Design

- Cache Addresses
  - Logical
  - Physical
- Cache Size
- Mapping Function
  - Direct
  - Associative
  - Set Associative
- Replacement Algorithm
- Write Policy
  - Write through
  - Write back
- Line Size
- Number of caches
  - Single or two level
  - Unified or split

Cache Addressing

- Where does cache sit?

— Between processor and virtual memory management unit

— Between MMU and main memory

- Logical cache (virtual cache) stores data using virtual addresses

— Processor accesses cache directly, not thorough physical cache

— Cache access faster, before MMU address translation

— Virtual addresses use same address space for different applications

– Must flush cache on each context switch

- Physical cache stores data using main memory physical addresses

Size Does Matter

- Cost

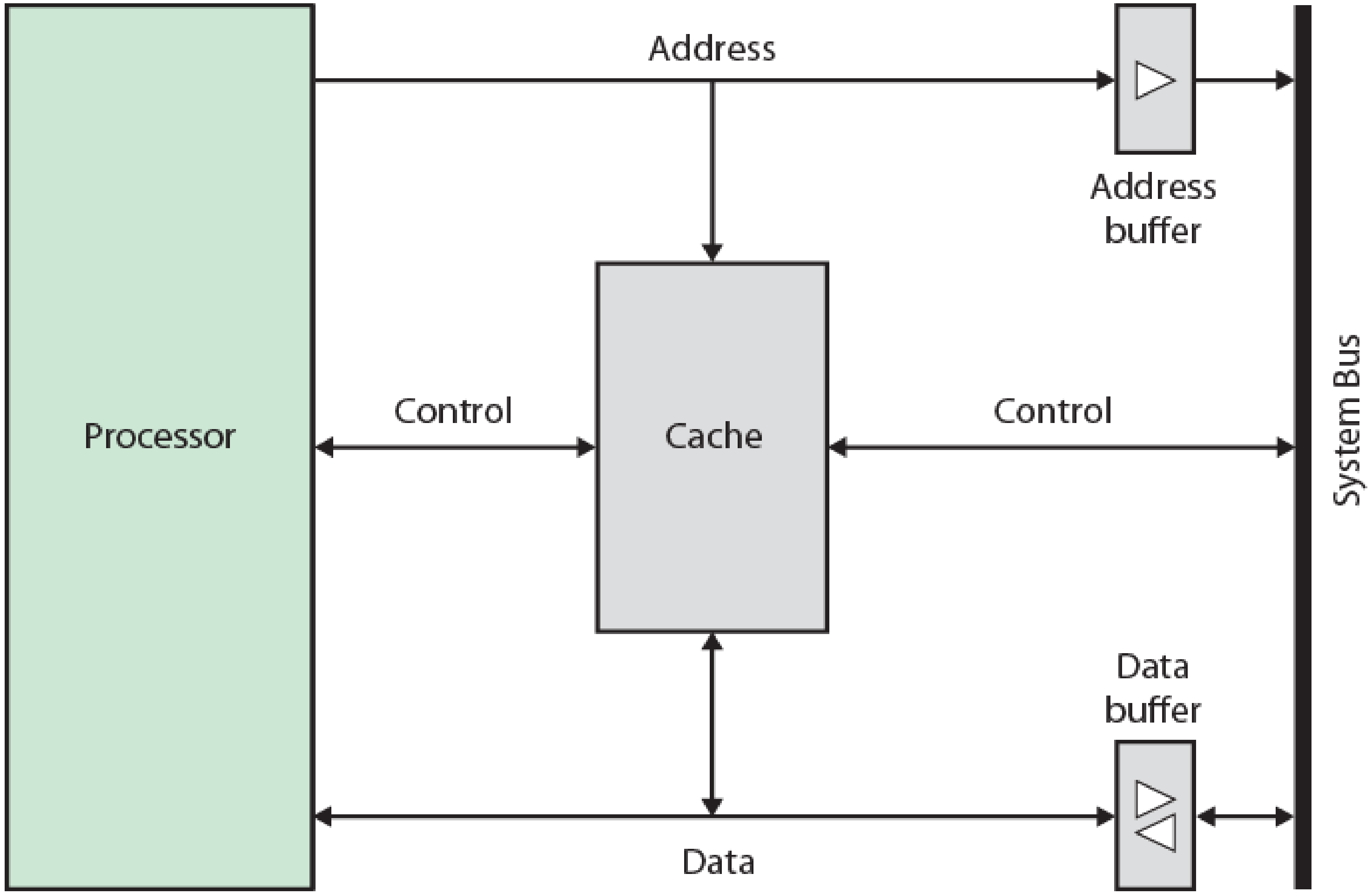
—More cache is expensive

- Speed

—More cache is faster (up to a point)

—Checking cache for data takes time

Typical Cache Organization



Comparison of Cache Sizes

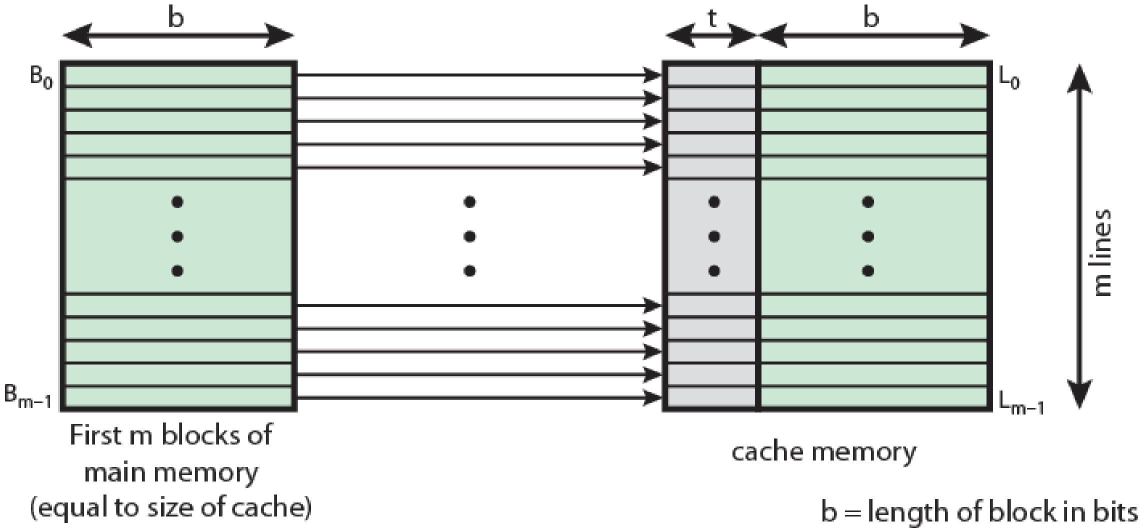
Processor	Type	Year of Introduction	L1 cache	L2 cache	L3 cache
IBM 360/85	Mainframe	1968	16 to 32 KB	—	—
PDP-11/70	Minicomputer	1975	1 KB	—	—
VAX 11/780	Minicomputer	1978	16 KB	—	—
IBM 3033	Mainframe	1978	64 KB	—	—
IBM 3090	Mainframe	1985	128 to 256 KB	—	—
Intel 80486	PC	1989	8 KB	—	—
Pentium	PC	1993	8 KB/8 KB	256 to 512 KB	—
PowerPC 601	PC	1993	32 KB	—	—
PowerPC 620	PC	1996	32 KB/32 KB	—	—
PowerPC G4	PC/server	1999	32 KB/32 KB	256 KB to 1 MB	2 MB
IBM S/390 G4	Mainframe	1997	32 KB	256 KB	2 MB
IBM S/390 G6	Mainframe	1999	256 KB	8 MB	—
Pentium 4	PC/server	2000	8 KB/8 KB	256 KB	—
IBM SP	High-end server/ supercomputer	2000	64 KB/32 KB	8 MB	—
CRAY MTAb	Supercomputer	2000	8 KB	2 MB	—
Itanium	PC/server	2001	16 KB/16 KB	96 KB	4 MB
SGI Origin 2001	High-end server	2001	32 KB/32 KB	4 MB	—
Itanium 2	PC/server	2002	32 KB	256 KB	6 MB
IBM POWER5	High-end server	2003	64 KB	1.9 MB	36 MB
CRAY XD-1	Supercomputer	2004	64 KB/64 KB	1MB	—

Direct Mapping

- **Direct mapping** maps each block of main memory into only one possible cache line.

- In direct mapping cache, instead of storing total address information with data in cache only part of address bits is stored along with data.

- The new data has to be stored only in a specified cache location as per the mapping rule for direct mapping. So it doesn't need replacement algorithm.



b = length of block in bits
t = length of tag in bits

(a) Direct mapping

Direct Mapping

Advantages:

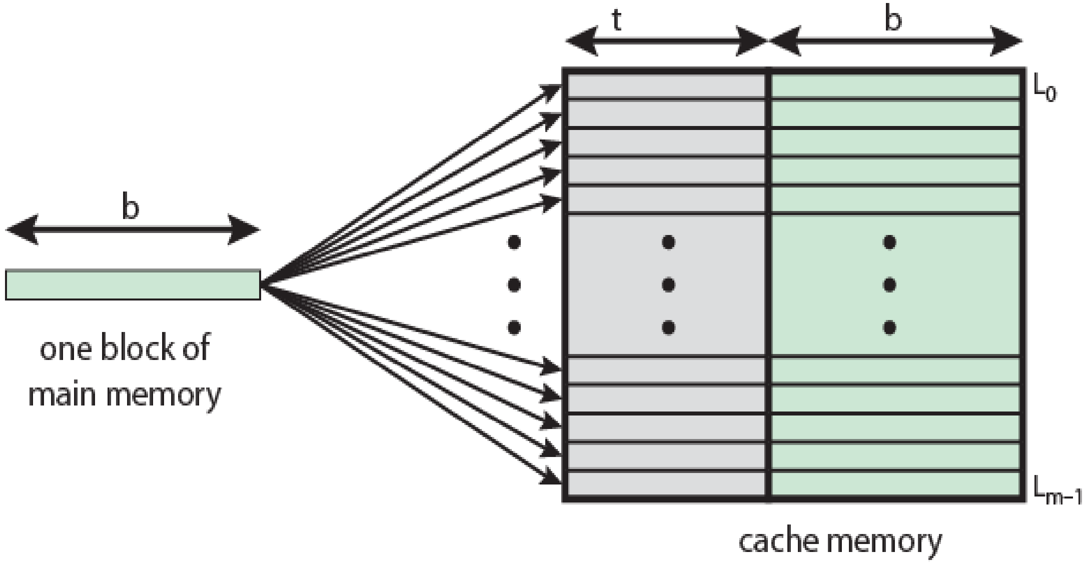
- Direct mapping is simplest type of cache memory mapping.
- Here only tag field is required to match while searching word that is why it fastest cache.
- Direct mapping cache is less expensive compared to associative cache mapping.

Disadvantage:

- The performance of direct mapping cache is not good as requires replacement for data-tag value.

Associative Mapping

- **Associative mapping** permits each main memory block to be loaded into any line of the cache.
- In *associative mapping* both the address and data of the memory word are stored.
- The associative mapping method used by cache memory is very flexible one as well as very fast.
- This mapping method is also known as ***fully associative cache***.



Associative Mapping

Advantages:

- Associative mapping is fast.
- Associative mapping is easy to implement.

Disadvantage:

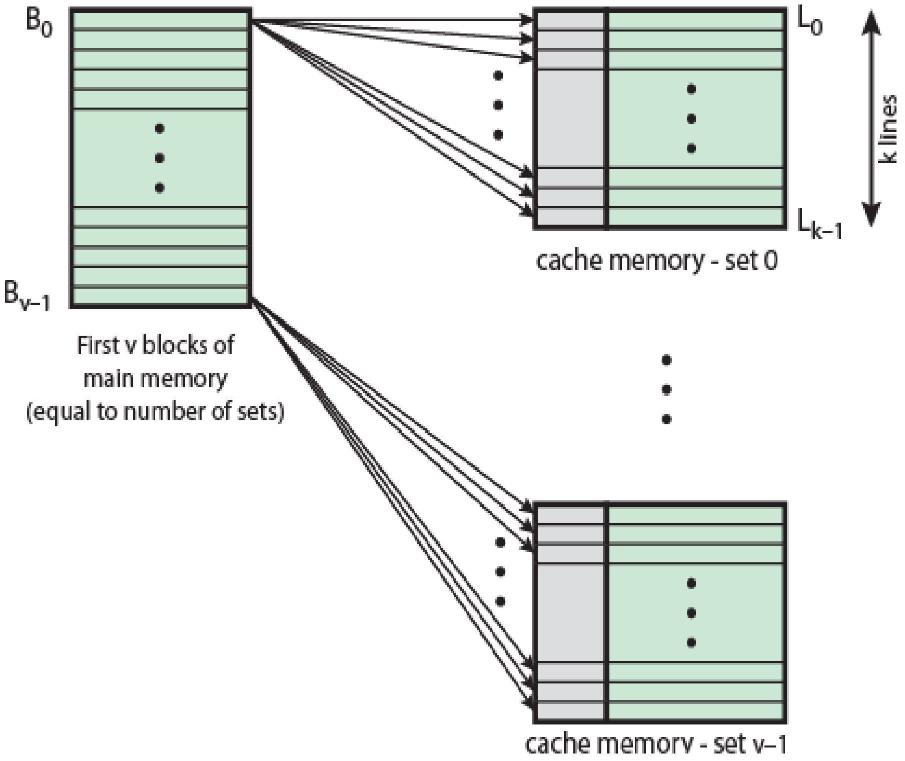
- Cache Memory implementing associative mapping is expensive as it requires to store address along with the data.

Set-Associative Mapping

Set-associative mapping, the cache is divided into a number of sets of cache lines; each main memory block can be mapped into any line in a particular set.

- In *Set-Associative cache memory* two or more words can be stored under the same index address.

- Here every data word is stored along with its tag. The number of tag-data words under an index is said to form a text.



Associative Mapping

Advantages:

- Set-Associative cache memory has highest hit-ratio compared two previous two cache memory discussed above. Thus its performance is considerably better.

Disadvantage:

- Set-Associative cache memory is very expensive. As the set size increases the cost increases.

Replacement Algorithms

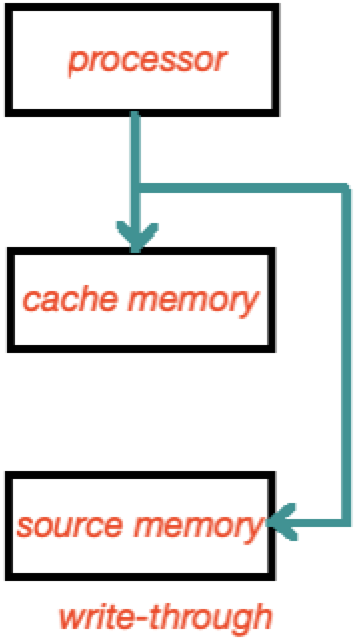
- Least recently used (LRU)
- First in first out (FIFO)
- Least frequently used (LFU)
- Random

Write Policy

- Must not overwrite a cache block unless main memory is up to date
- Multiple CPUs may have individual caches
- I/O may address main memory directly

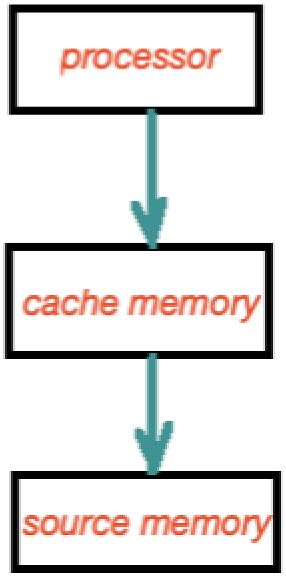
Write Through

- All writes go to main memory as well as cache
- Multiple CPUs can monitor main memory traffic to keep local (to CPU) cache up to date
- Lots of traffic
- Slows down writes



Write Back

- Updates initially made in cache only
- Update bit for cache slot is set when update occurs
- If block is to be replaced, write to main memory only if update bit is set
- Other caches get out of sync
- I/O must access main memory through cache
- N.B. 15% of memory references are writes



write-back

Line Size

- Retrieve not only desired word but a number of adjacent words as well

- Increased block size will increase hit ratio at first

— the principle of locality

- Hit ratio will decreases as block becomes even bigger

— Probability of using newly fetched information becomes less than probability of reusing replaced

- Larger blocks

— Reduce number of blocks that fit in cache

— Data overwritten shortly after being fetched

— Each additional word is less local so less likely to be needed

Multilevel Caches

- High logic density enables caches on chip

—Faster than bus access

—Frees bus for other transfers

- Common to use both on and off chip cache

—L1 on chip, L2 off chip in static RAM

—L2 access much faster than DRAM or ROM

—L2 often uses separate data path

—L2 may now be on chip

—Resulting in L3 cache

– Bus access or now on chip…

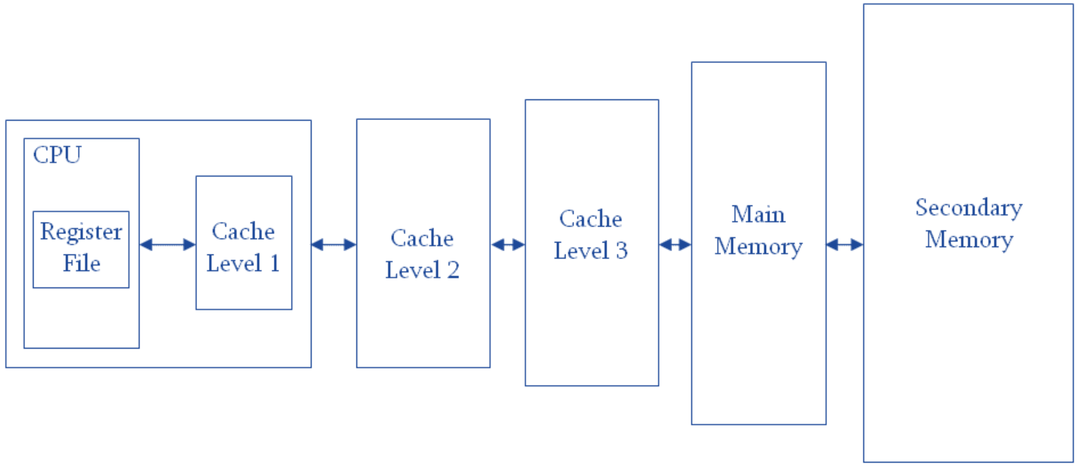


Figure : Conceptual Organization of Multilevel Memories in a Computer System

Unified v Split Caches

- *One cache for data and instructions* or *two, one for data and one for instructions*

- **Advantages of unified cache**

–Higher hit rate

–Balances load of instruction and data fetch

–Only one cache to design & implement

- **Advantages of split cache**

–Eliminates cache contention between instruction fetch/decode unit and execution unit

–Important in pipelining