Memory

- A physical device that is used to store data and program (sequence of instructions) on a temporary or permanent basis for performing specific tasks on the computer.
- There are basically two types of memory:
  - Internal memory (main memory, registers, cache)
  - External memory (disk, tape)
- For internal memory, the **capacity** is expressed in terms of bytes or words. Common word lengths are 8 bits, 16 bits or 32 bits.
- For external memory, **capacity** is expressed in terms of bytes.
- For internal memory, the **unit of transfer** (number of bits read out or written into the memory at a time) is the number of electrical lines into and out of the memory; may equal to the word length or addressable units.
- For external memory, data are often transferred (**unit of transfer**) in much larger units than a word, referred to as **blocks**.

See Stallings (Chapter: Cache Memory)

Memory

- **Access time (latency)**: For main memory (RAM), the time a processor takes to perform a read or write operation. For non-RAM memory, the time it takes to position the read-write mechanism at the desired location. Access time directly affects how fast the computer processes data.
- **Cycle time**: For main memory (RAM), it is the access time plus an additional time before the next access is started. It is measured in nanoseconds. It is concerned with the system bus, not with the processor.
- **Transfer rate**: The rate at which data can be transferred into or out of a memory. For, main memory (RAM), it is equal to 1/cycle time. For non-RAM memory, it is measured as follows:

  $T_n = T_a + n/R$

  where

  T_n = average time to read or write n bits

  T_a = average access time

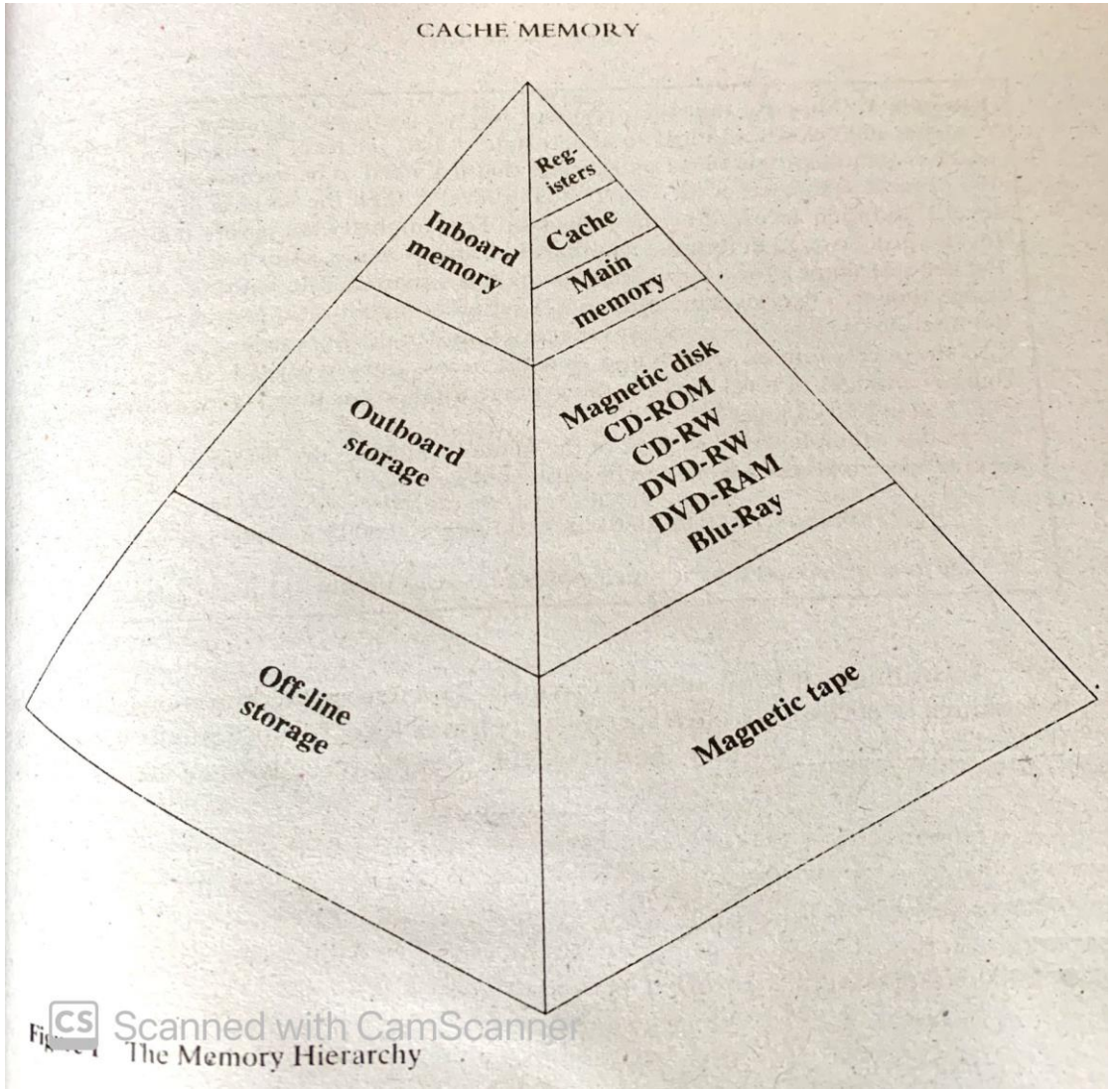
  n = number of bits

  R = transfer rate in bits per seconds

Memory Hierarchy

- Three key points are very important when designing a memory; **capacity**, **access time**, and **cost**.
- Faster access time results more cost per bit, larger capacity results lesser cost per bit, and larger capacity results slower access time as well.
- To get out of this dilemma is not to rely on a single memory component or technology, but to employ a **memory hierarchy**.
- In the memory hierarchy figure, the following occurs as goes down:
  - Decreasing cost per bit
  - Increasing capacity
  - Increasing access time
  - Decreasing the frequency of access of the memory by the processor.

smaller, more expensive and faster



The Memory Hierarchy

Decreasing the frequency of access

Memory Hierarchy

- It is possible to organize data across the hierarchy such that percentage of accesses to each successively lower level is substantially less than that of the level above.
- The fastest, smallest, and most expensive memory is **registers**, which are local/internal to the processor.
- **Main memory** is the principal internal memory of the computer.
- Each location in the main memory has a unique address.
- Registers, main memory, and cache are volatile and employ semiconductor technology.
- A portion of the main memory can be used as **buffer** to hold data temporarily for being processed.
- The **external non-volatile memory** is referred to as secondary/auxiliary memory or mass storage device (hard disk, removable disk).
- These are used to store programs and data files and visible to programmers.
- **Virtual memory** is a mechanism when the RAM is low and the portion of the secondary storage acts as the extension of the main memory.

Cache Memory

- Cache is a special, very high-speed memory and smaller memory.
- The main memory is extended with cache.
- It transfers data between main memory and the processor.
- It is used to speed up and synchronizing with the high-speed CPU.
- It acts as a buffer between RAM and the CPU.
- It holds frequently requested data and instructions so that they are immediately available to the CPU when needed.

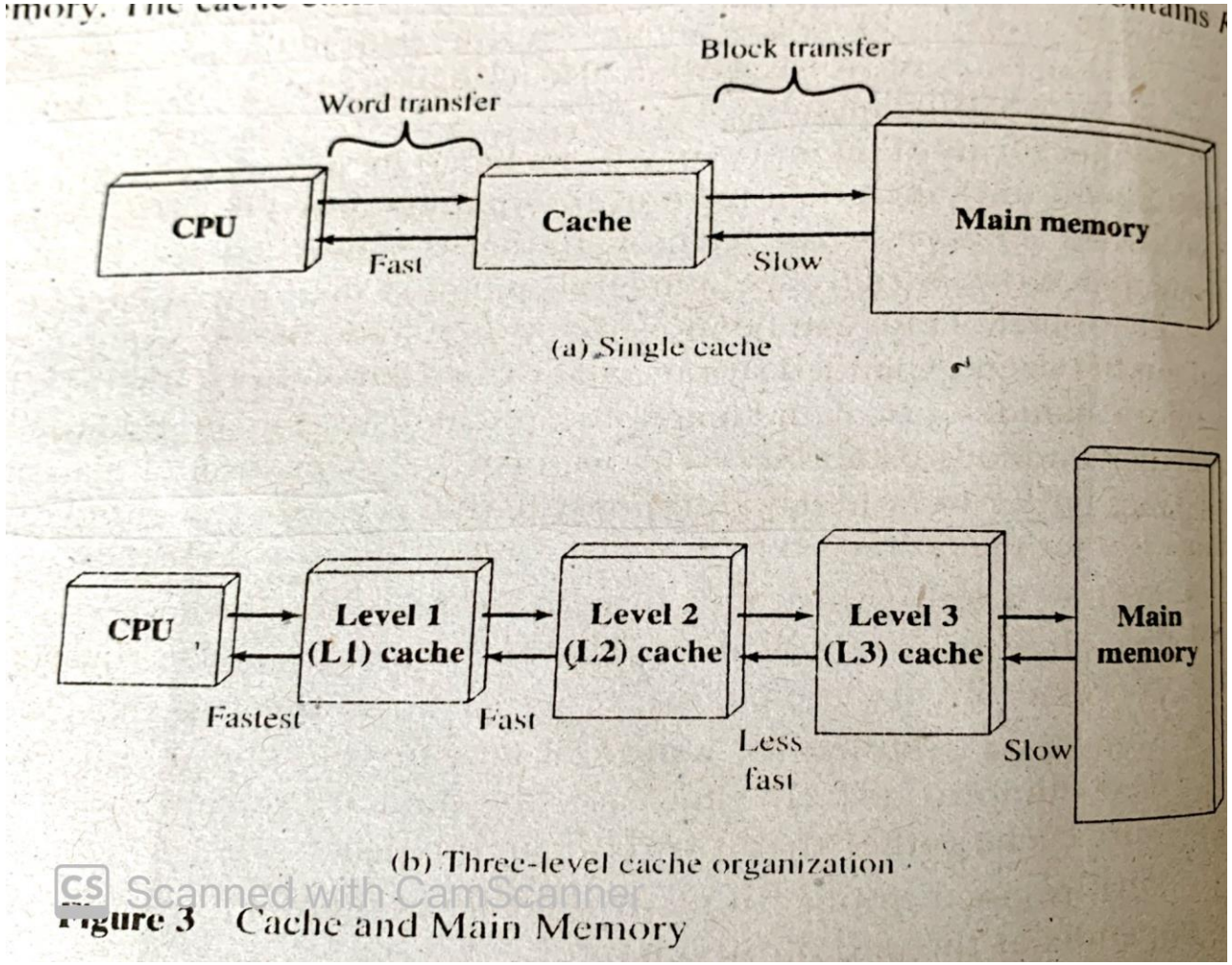


Figure 3 Cache and Main Memory

Structure of Cache and Main Memory

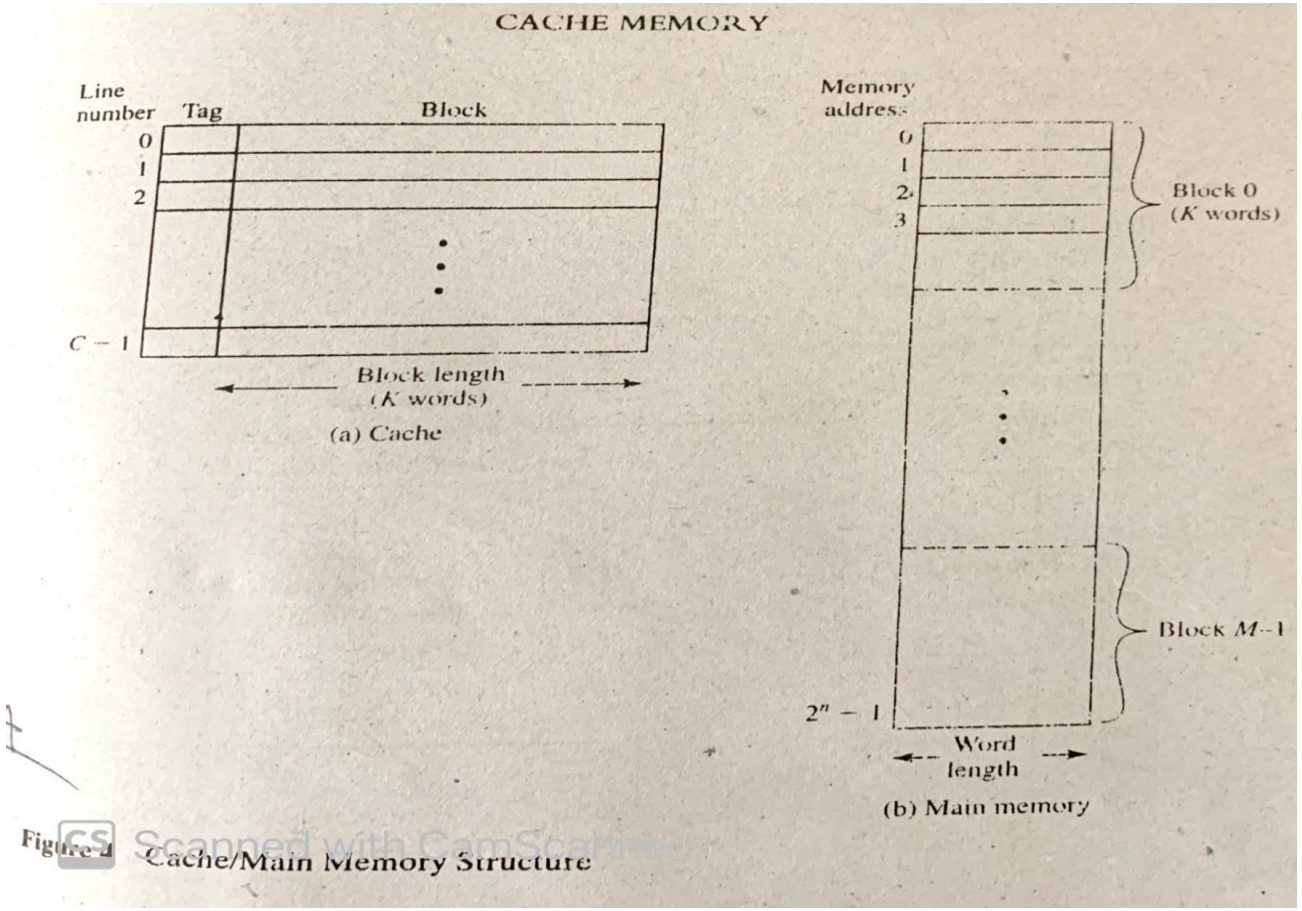


Figure 4 Cache/Main Memory Structure

Cache Read Operation

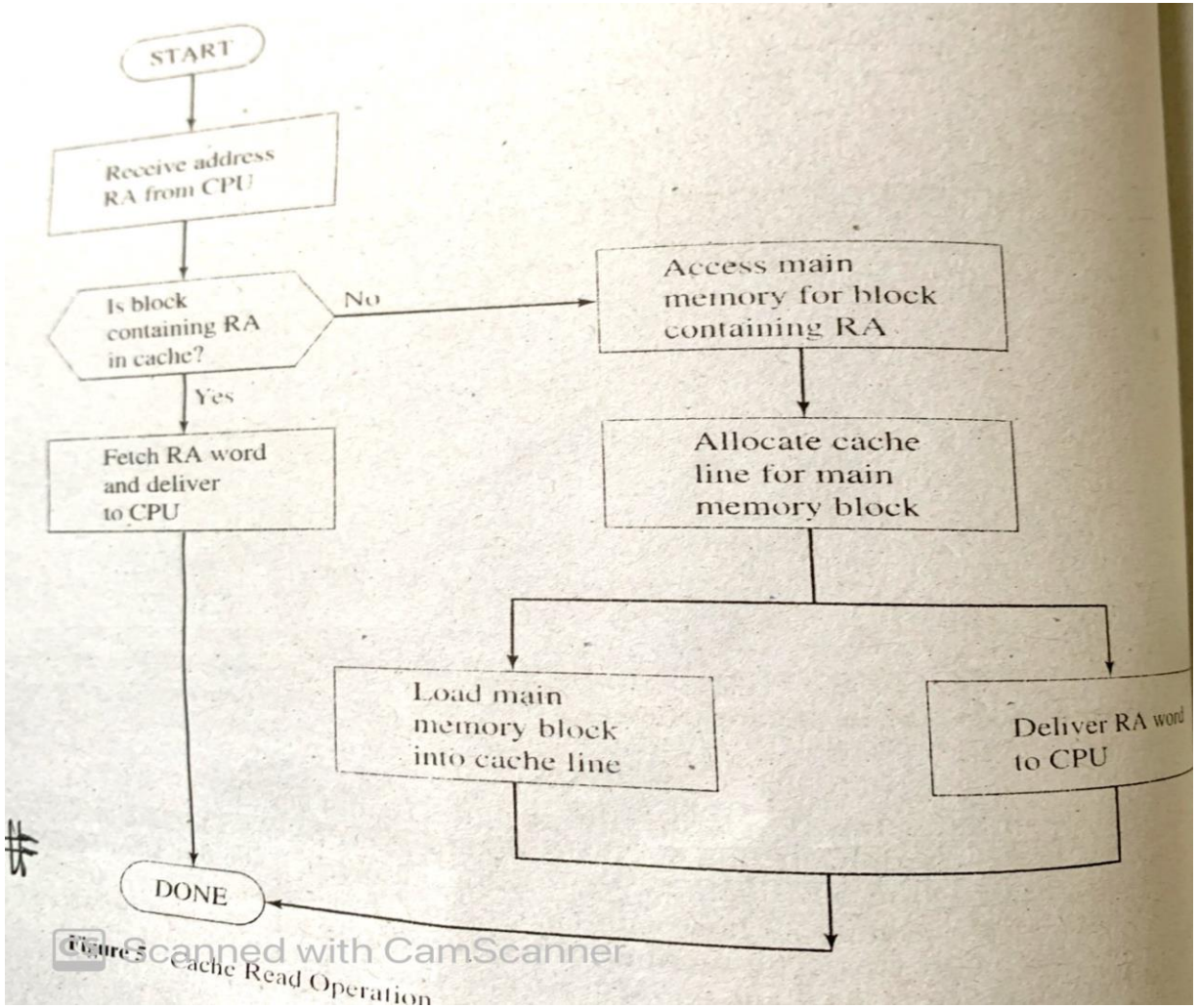


Figure 5 Cache Read Operation

Logical Address and Physical Address

- The system designer may choose the place of cache.

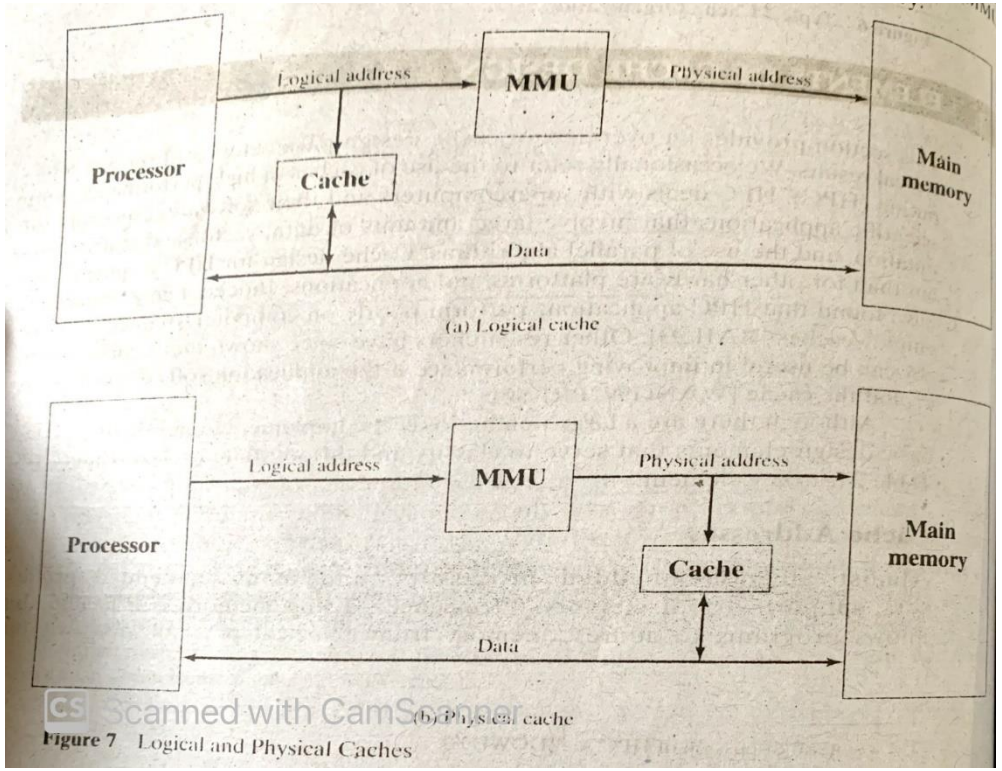


Figure 7 Logical and Physical Caches

Memory Mapping

- **Memory mapping** is the translation/transformation between the logical address space and the physical memory.
- The transformation of data from main memory to cache memory is called mapping.
- **Address** uniquely identifies a location of a memory.
- There are two types of address: **logical address** and **physical address**.
- Logical address is generated by the CPU while a program is running.
- Logical address is a virtual address. User can see it.
- Physical address is the actual location in the memory unit. User can not see it.
- Logical address is used as a reference to access the physical address.
- Physical address is computed by the memory management unit.

Memory Mapping

- There are fewer cache lines than main memory blocks.
- So an way/algorithm is required for mapping main memory blocks into cache lines.
- Also a way is required to determine which main memory blocks currently occupies a cache line.
- This is done through three types of mapping:
  - Direct mapping
  - Associative mapping
  - Set associative mapping

Direct Mapping

- This simplest technique maps each block of memory into only one possible cache line.
- This mapping is expressed as follows:

$$i = j \ modulo \ m$$

where

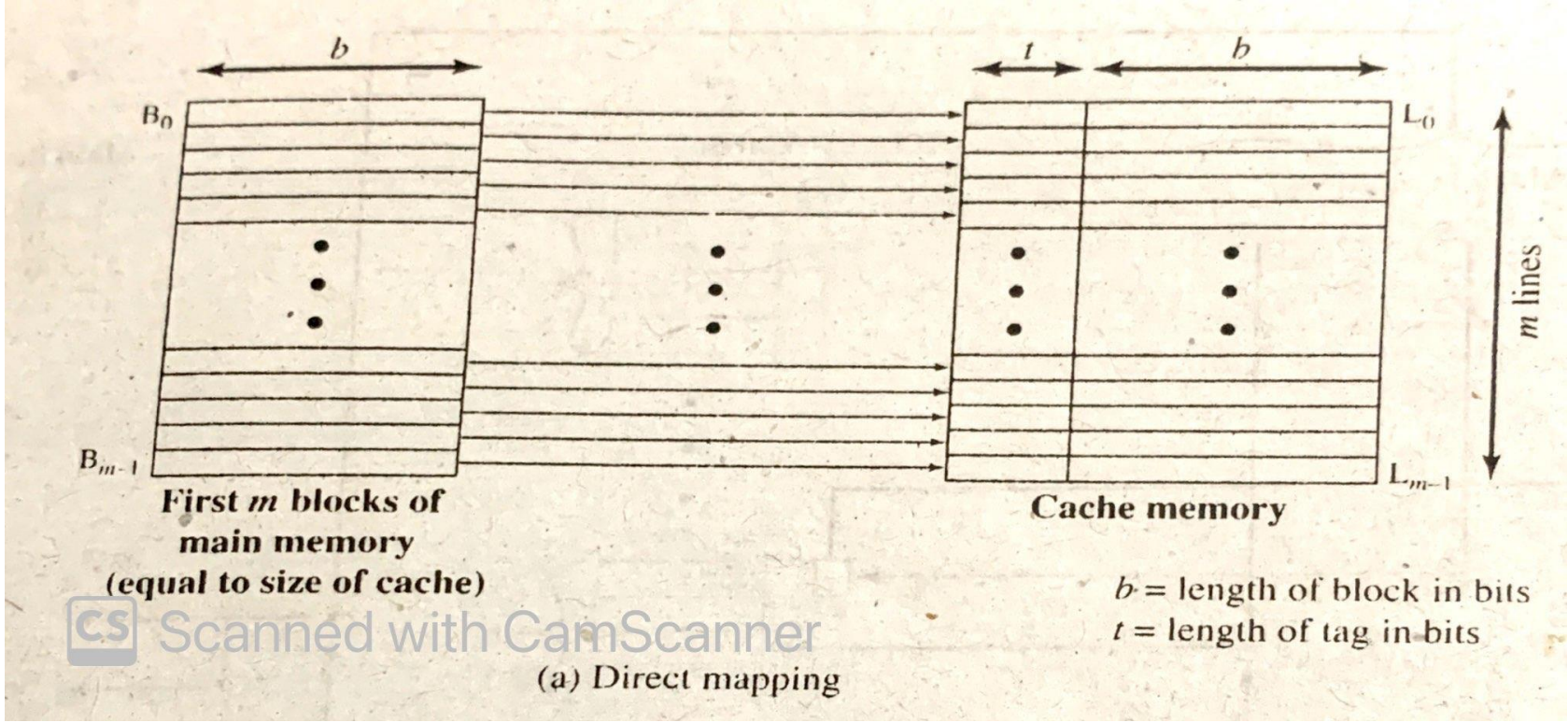
i = cache line number

j = main memory block number

m = number of lines in the cache

See Example 2

Direct Mapping



(a) Direct mapping

Direct Mapping

- This mapping is easily implemented using the main memory address.
- Direct mapping divides an address into three parts:
  - **word bits**: The word bits are the least significant bits that identify the specific word within a block of memory.
  - **line bits**: The line bits are the next least significant bits that identify the line of the cache in which the block is stored.
  - **tag bits**: The remaining bits are stored along with the block as the tag which locates the block's position in the main memory.
- Thus the use of a portion of the address as a line number provides unique mapping of each block of main memory into the cache.
- When a block is actually read into an assigned cache line, it is necessary to tag the data to distinguish it from other blocks that can fit into that line.
- This technique is simple and inexpensive.