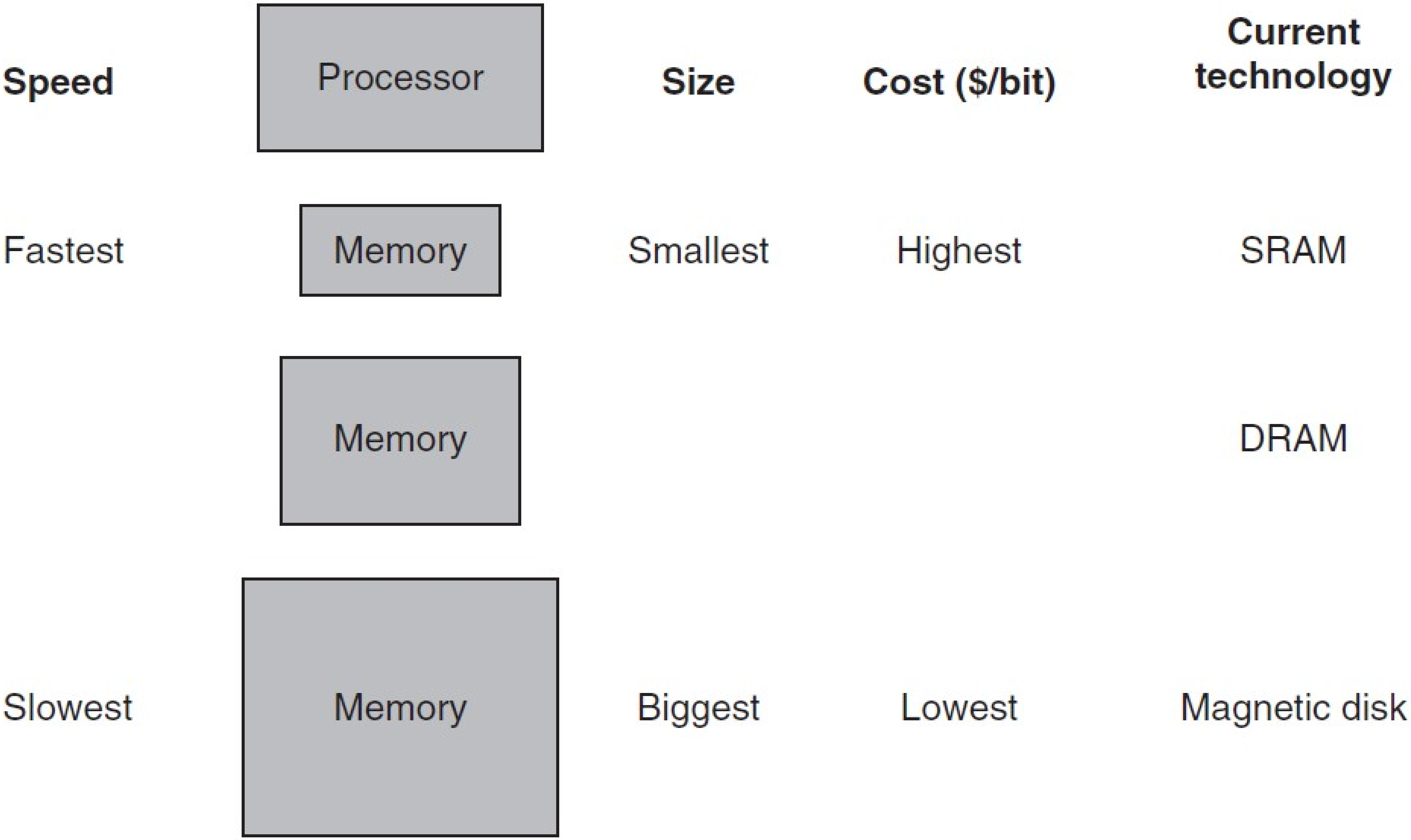
Memory Hierarchy

Chapter 5. Patterson and Hennessy. 4ed.
Chapter 6. JPH.

Introduction

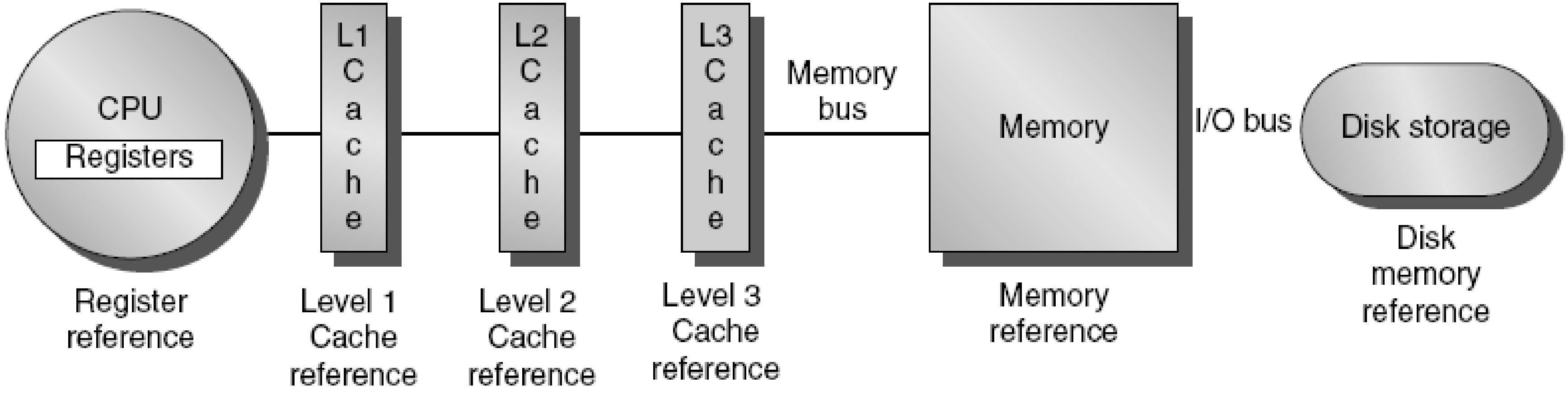
- Unlimited amounts of memory with low latency
- Memory latency is large
- Fast memory technology is more expensive per bit than slower memory
- Solution: organize memory system into a hierarchy
  - Entire addressable memory space available in largest, slowest memory
  - Incrementally smaller and faster memories, each containing a subset of the memory below it, proceed in steps up toward the processor

Memory Hierarchy



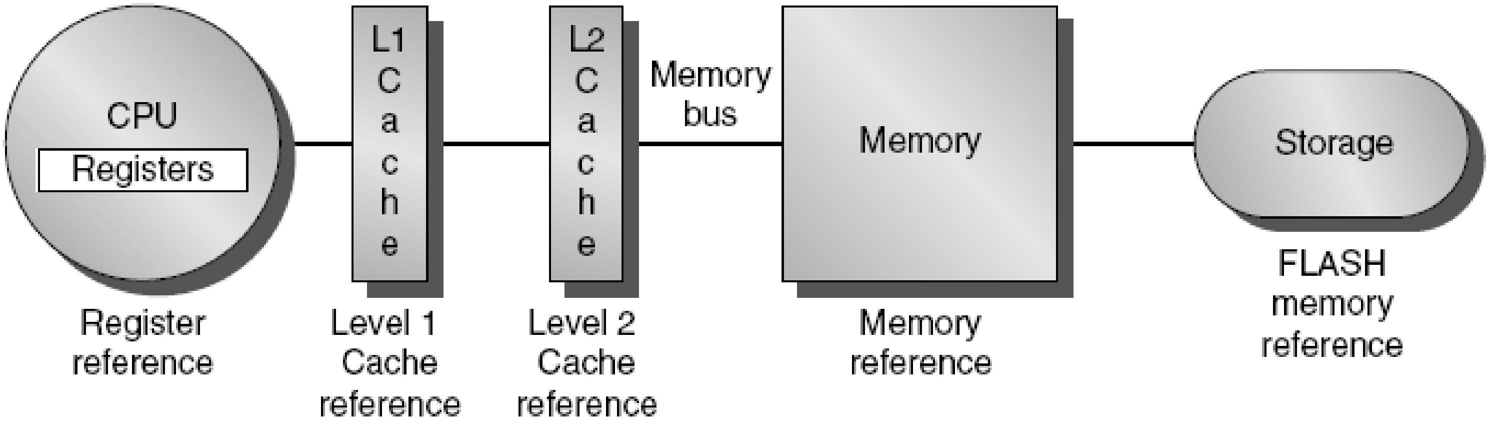
Speed	Processor	Size	Cost ($/bit)	Current technology
Fastest	Memory	Smallest	Highest	SRAM
	Memory			DRAM
Slowest	Memory	Biggest	Lowest	Magnetic disk

The Memory Hierarchy



	Register reference	Level 1 Cache reference	Level 2 Cache reference	Level 3 Cache reference	Memory reference	Disk memory reference
Size:	1000 bytes	64 KB	256 KB	2–4 MB	4–16 GB	4–16 TB
Speed:	300 ps	1 ns	3–10 ns	10–20 ns	50–100 ns	5–10 ms

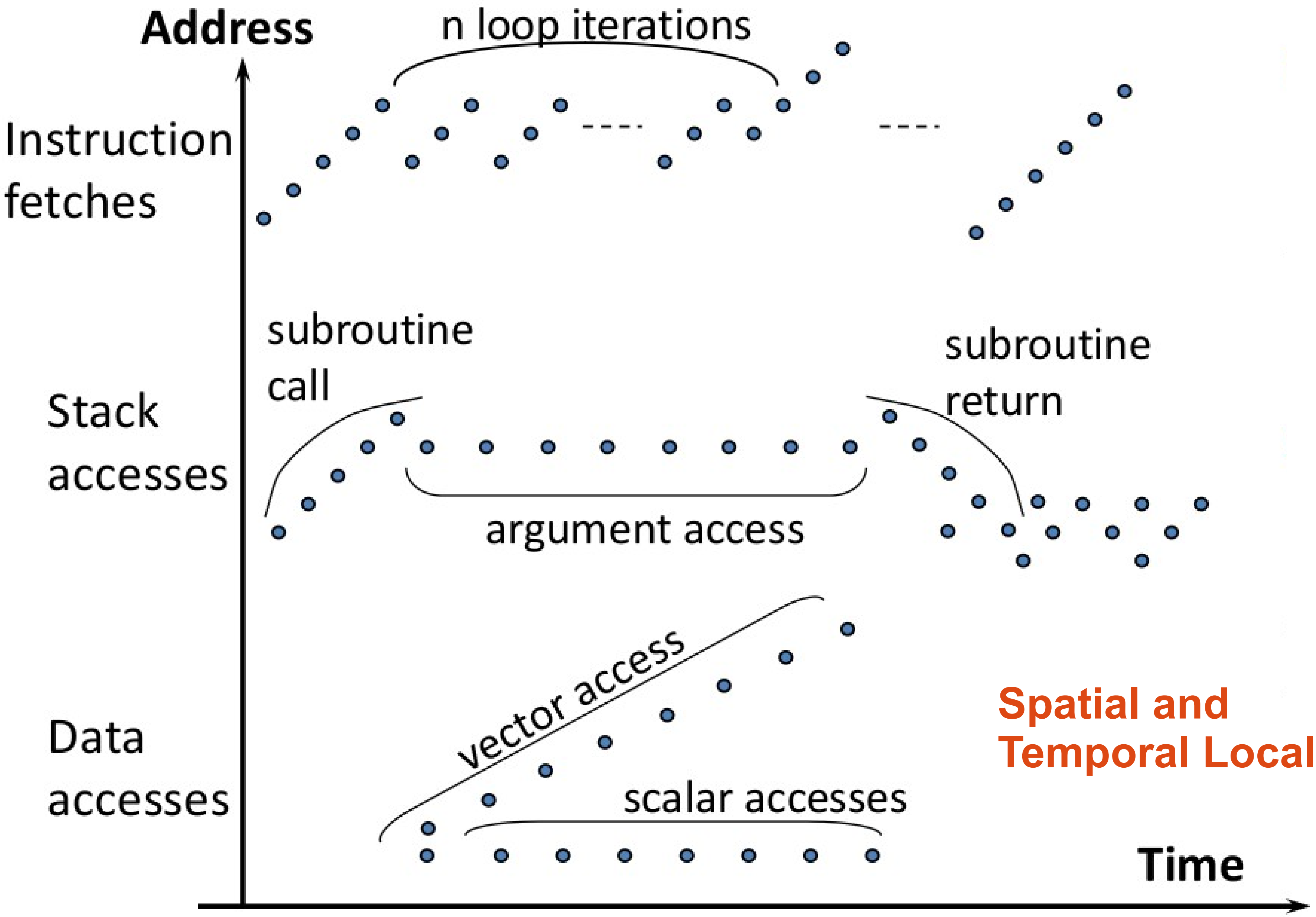
(a) Memory hierarchy for server



	Register reference	Level 1 Cache reference	Level 2 Cache reference	Memory reference	FLASH memory reference
Size:	500 bytes	64 KB	256 KB	256–512 MB	4–8 GB
Speed:	500 ps	2 ns	10–20 ns	50–100 ns	25–50 us

(b) Memory hierarchy for a personal mobile device

Predictable Memory Reference Patterns



Spatial and Temporal Locality

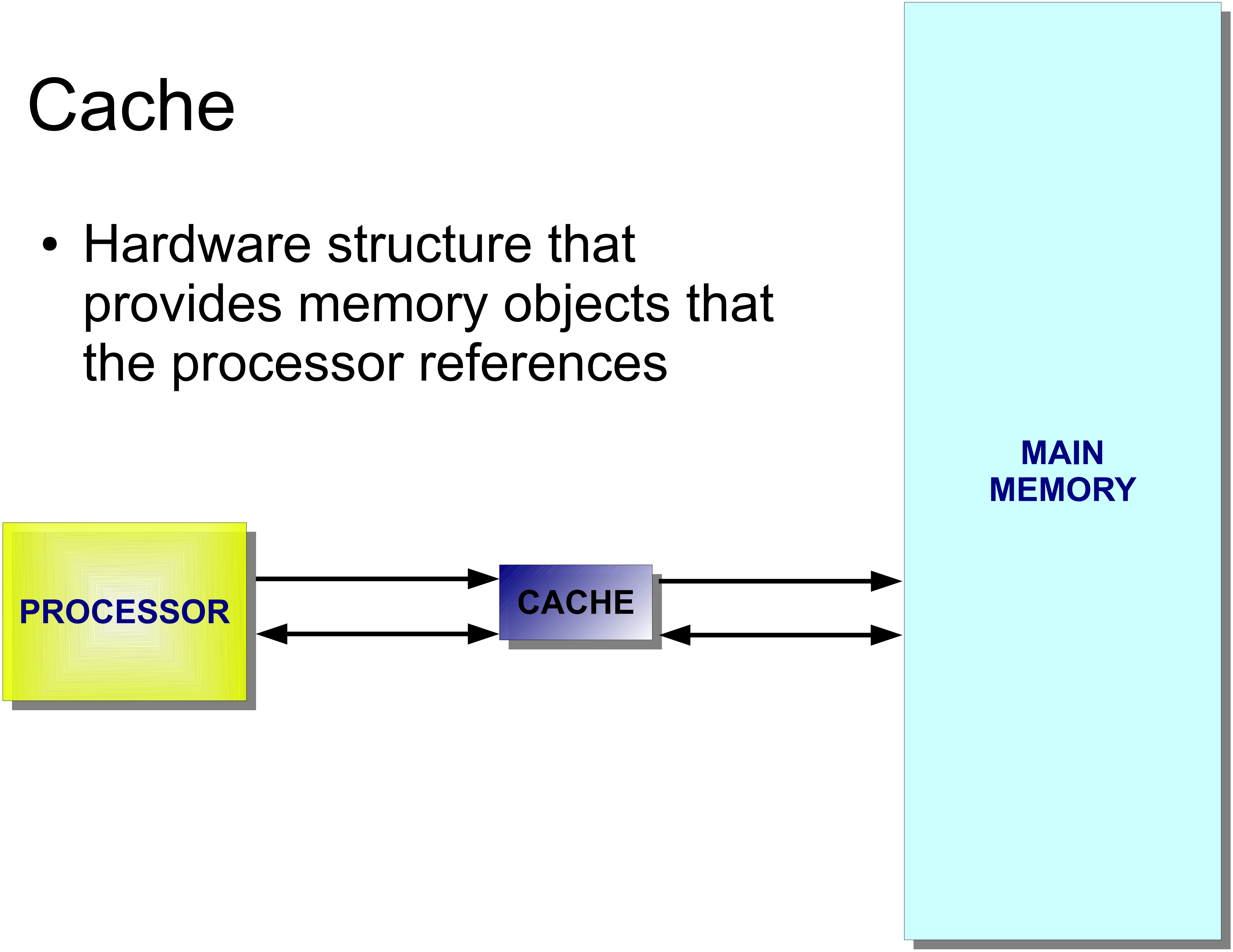
Hatfield and Gerald: Program Restructuring for Virtual Memory. IBM Systems Journal 10(3): 168-192 (1971)

Locality of Reference

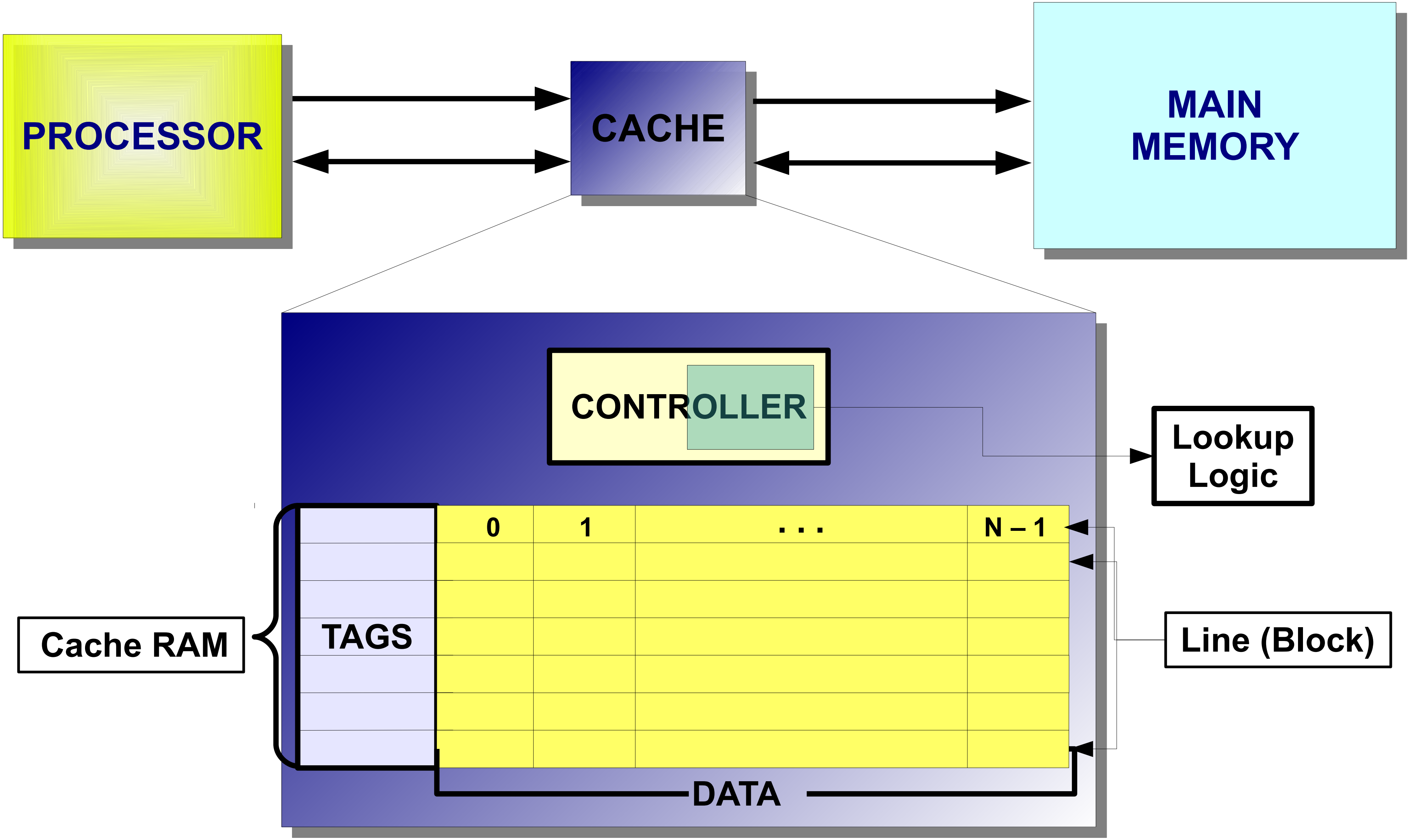
- Temporal locality (locality in time): if an item is referenced, it will tend to be referenced again soon.
- Spatial locality (locality in space): if an item is referenced, items whose addresses are close by will tend to be referenced soon.

Cache

- Hardware structure that provides memory objects that the processor references



Inside a Cache



PROCESSOR

CACHE

MAIN MEMORY

CONTROLLER

Lookup Logic

Cache RAM

TAGS

0

1

. . .

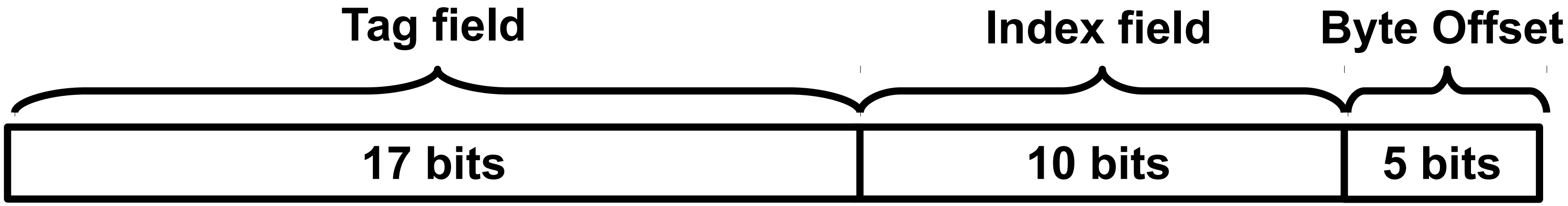
N – 1

Line (Block)

DATA

Direct Mapped Cache

- 32 bit address. Byte addressable memory. Cache Block size = 32B. Cache size = 32KB. What is the Index field size? What is the Tag field size?



Direct Mapped Cache Organization

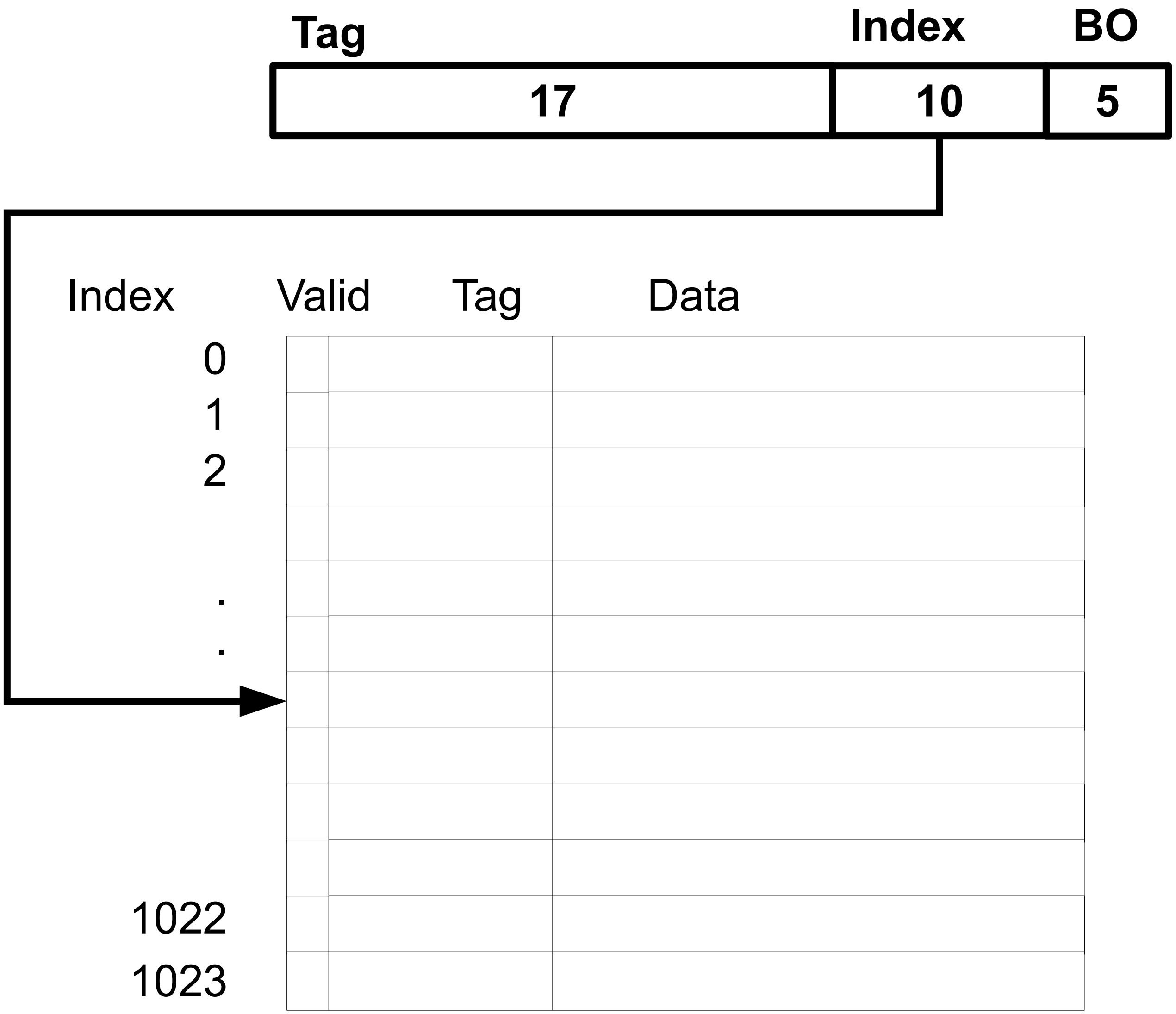
Index	Valid	Tag	Data
0			
1			
2			
.			
.			
.			
1022			
1023			

Direct Mapped Cache Organization

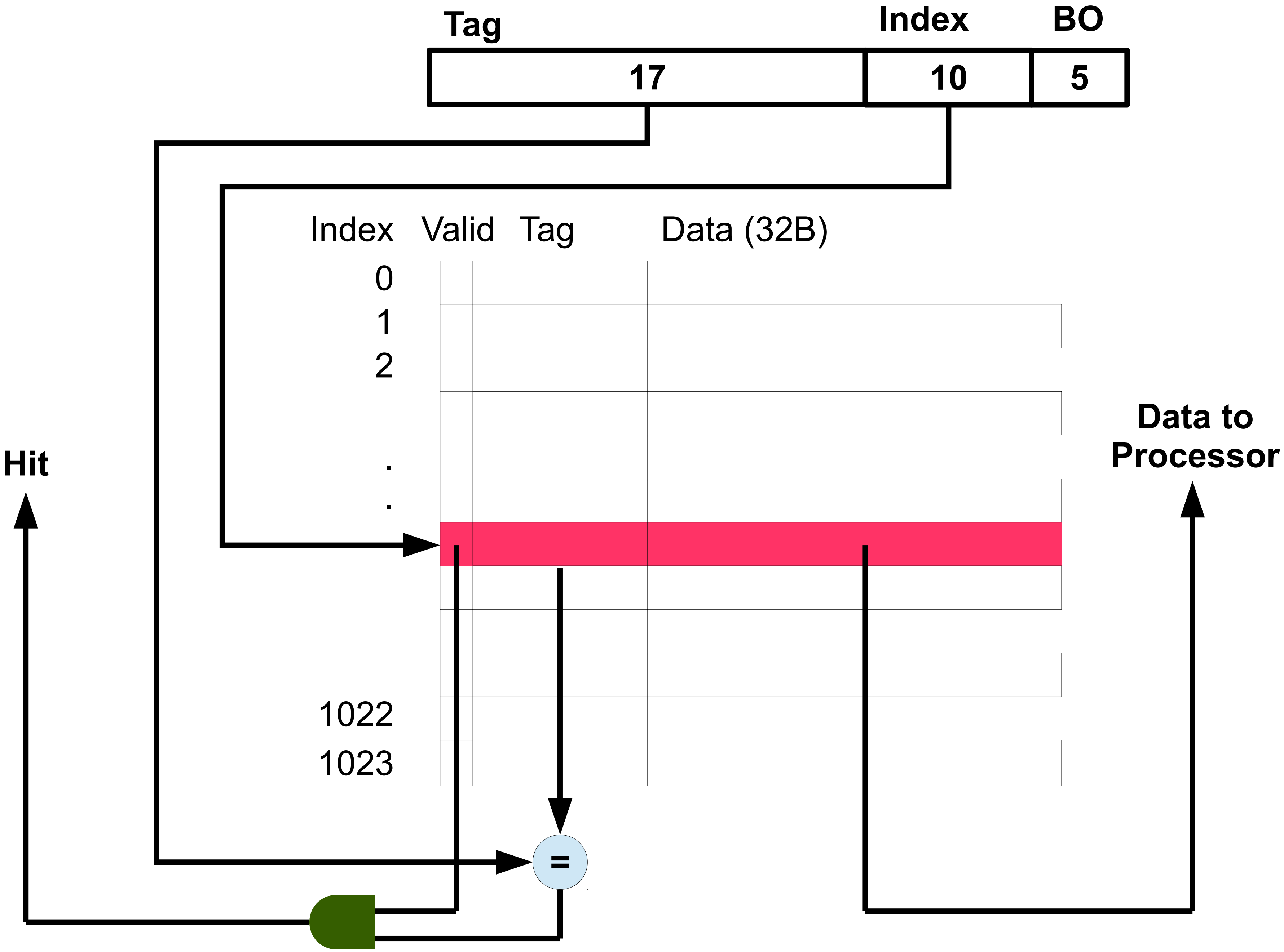
32 bit address from Processor

Index	Valid	Tag	Data
0			
1			
2			
.			
.			
.			
1022			
1023			

Direct Mapped Cache Organization



Direct Mapped Cache Organization



Tag

Index

BO

17

10

5

Index

Valid

Tag

Data (32B)

0

1

2

.

.

1022

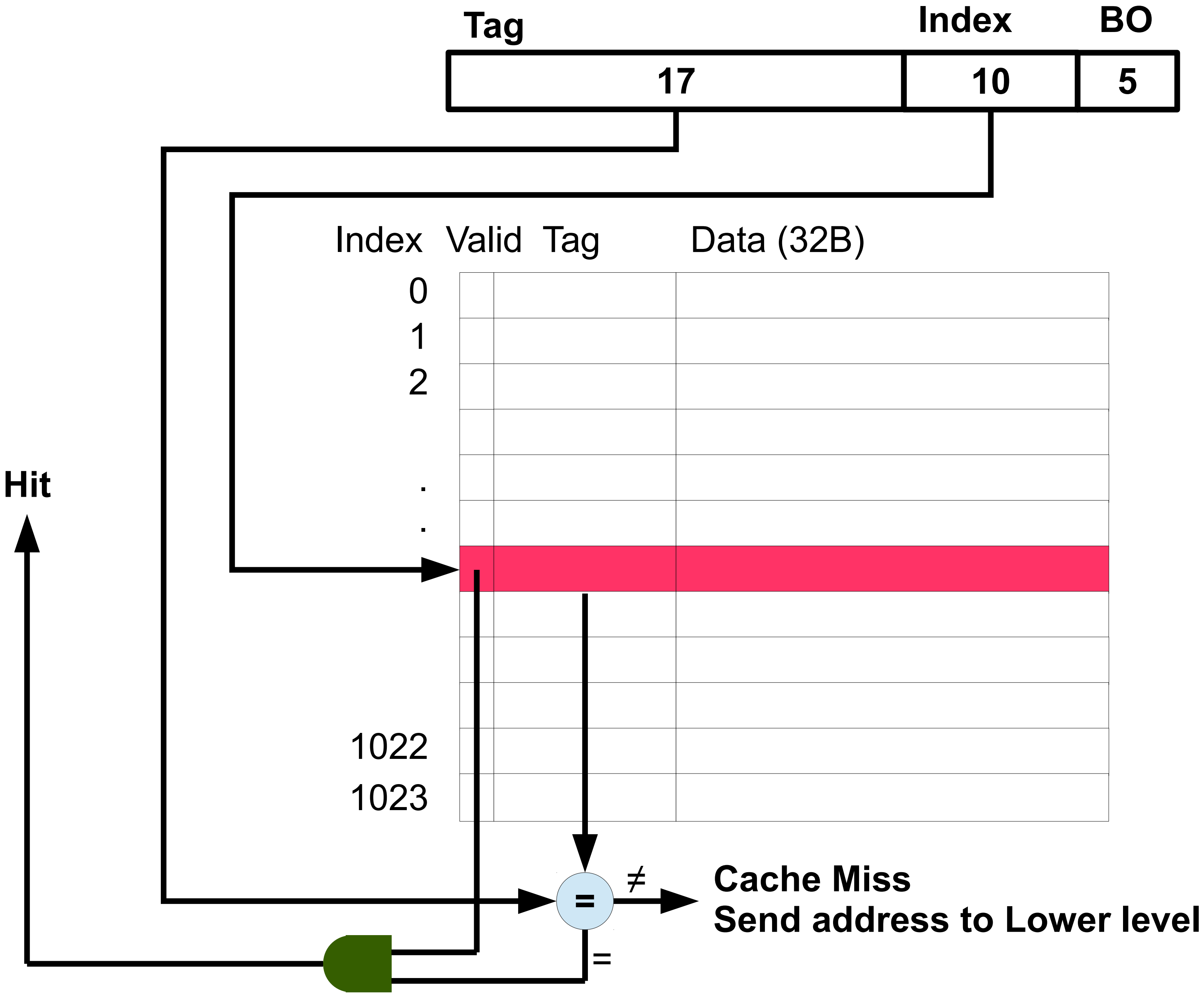
1023

=

Hit

Data to Processor

Direct Mapped Cache Organization



Tag

Index

BO

17

10

5

Index

Valid

Tag

Data (32B)

0

1

2

.

.

1022

1023

Hit

=

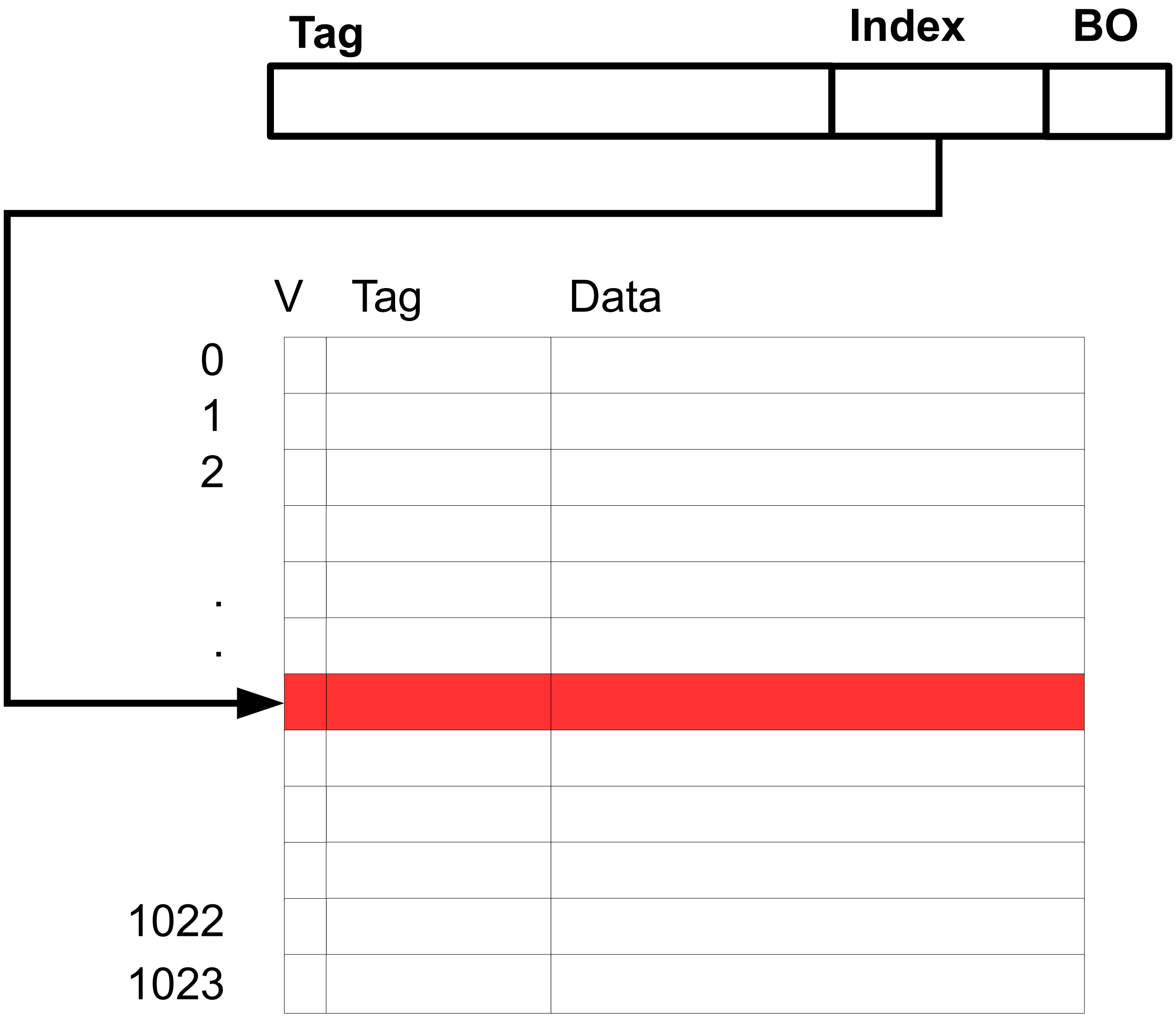
≠

=

Cache Miss

Send address to Lower level

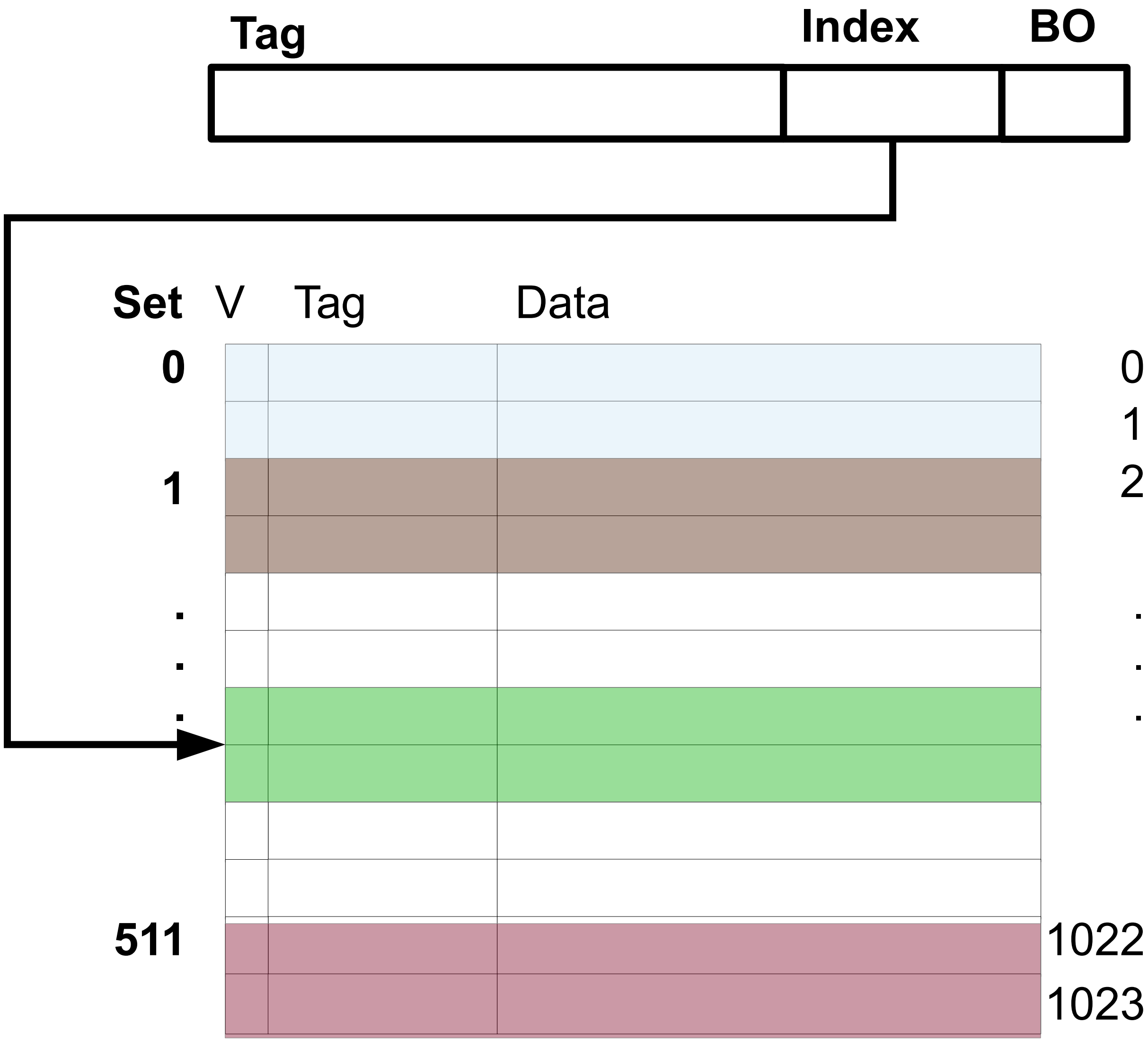
Block Placement



Direct Mapped Cache

Index bits Identify a unique Cache line

Block Placement



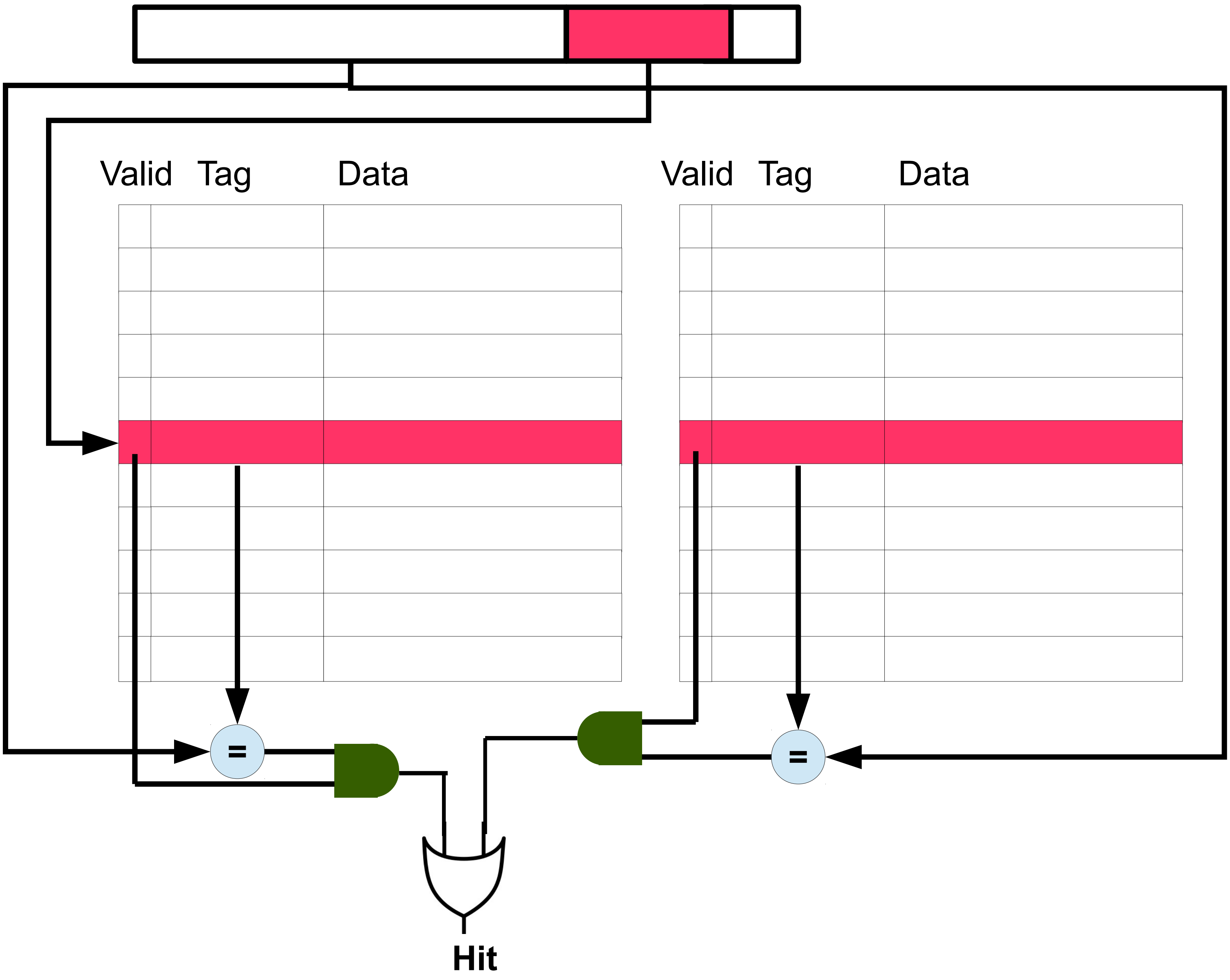
Set Associative Cache

Index bits Identify a unique SET

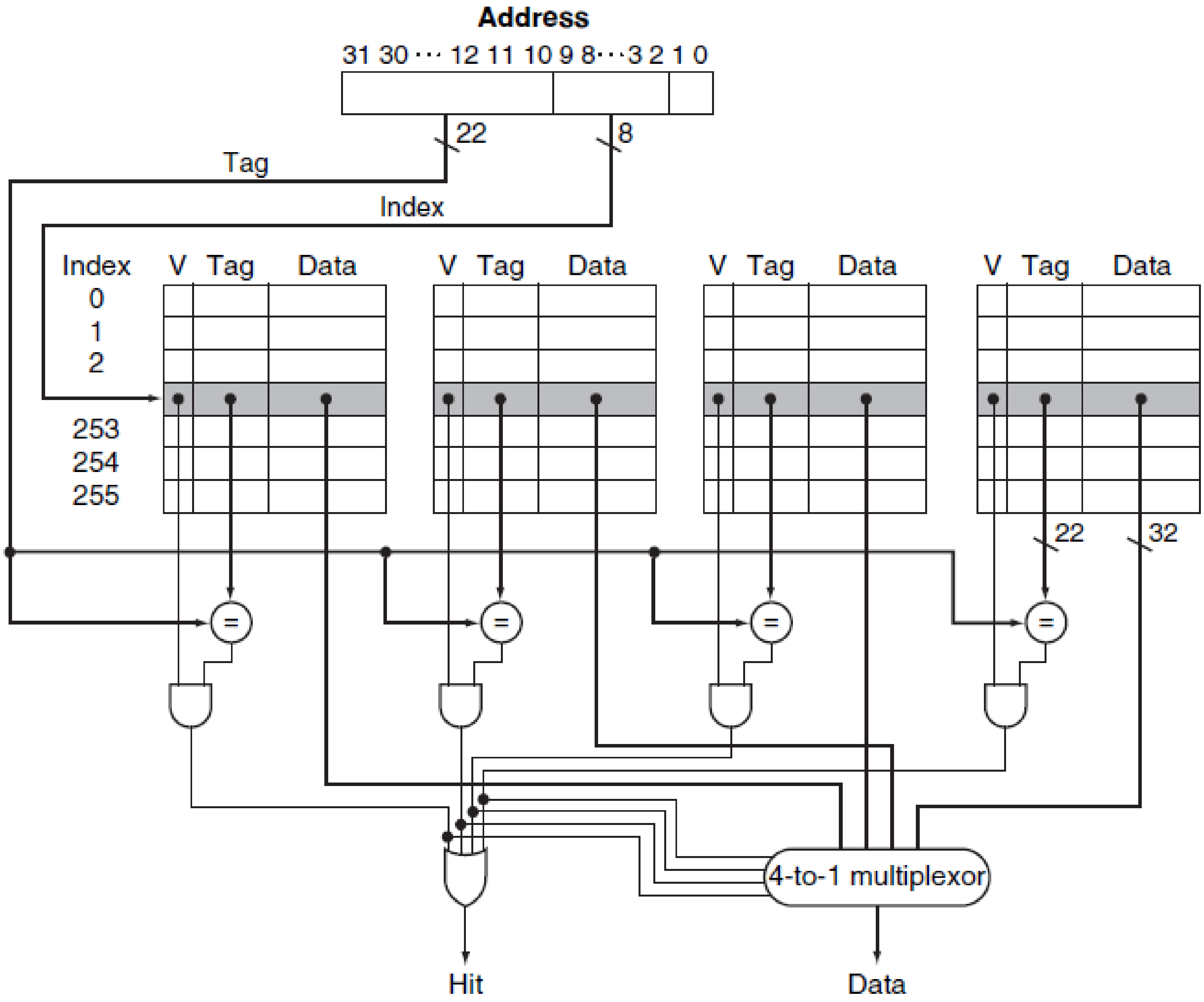
A Set contains multiple cache lines

2-way Set Associative Cache

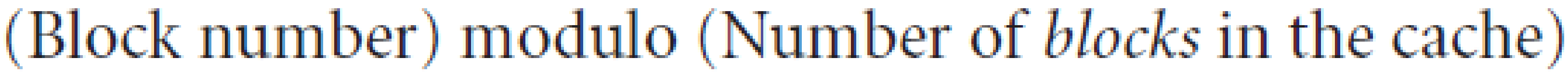
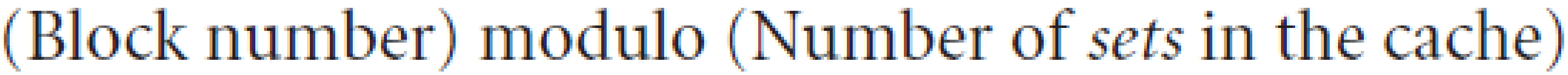
2–way Set Associative Cache



4-way Set Associative Cache



Block Placement

- Direct Mapped Cache
  - A block can be placed in exactly one location in the cache
  - 
    (Block number) modulo (Number of *blocks* in the cache)
- Fully Associative Cache
  - A block can be placed in any location in the cache
- Set Associative Cache
  - A block can be placed in any location inside a set in the cache
  - 
    (Block number) modulo (Number of *sets* in the cache)

Examples

Consider a cache with 64 blocks and a block size of 16 bytes. To what block number does byte address 1200 map?

- DM Cache
- 4-way SA Cache
- Fully SA Cache

What is the size of the Cache RAM (in bits)? 16 KB of data. 32-bit address.

1. Direct-mapped cache with 4-word blocks.
2. 2-way Set Associative cache with 8-word blocks
3. Fully Set Associative with 8-word blocks

Block Replacement

- Which block should be evicted when a new block is about to be fetched into the cache?
  - In Direct Mapped Cache?
- Least Recently Used (LRU)
- First in First Out (FIFO)
- Random Replacement Policy

Cache Writes

- On a Write Hit
  - When does the cache update the modified block in the lower level?
  - As soon as a write occurs: **Write through policy**.
    - Large stall time
    - Write buffer
  - When the block is replaced: **Write back policy**.
    - Multiple writes within a block require only one write to the lower level in the hierarchy.
- On a Write Miss
  - **No write-allocate**: write the data to memory only.
  - **Write-allocate**: read the entire block into the cache, then write the word

Four Memory Hierarchy Questions

- Block Placement
  - Where can a block be placed in a cache?
  - Direct mapped, Set associative, Fully associative
- Block Identification
  - How is a block found if it is in cache?
- Block Replacement
  - Which block should be replaced on a miss?
- Write Strategy
  - What happens on a write?

The Three Cs

- **Compulsory misses**
  - Caused by the first access to a block that has never been in the cache
  - Also called cold-start misses
- **Capacity misses**
  - Caused when the cache cannot contain all the blocks needed during execution of a program
  - Occur when blocks are replaced and then later retrieved.
- **Conflict misses**
  - Occur when multiple blocks compete for the same set
  - Conflict misses are eliminated in a fully associative cache of the same size as the SA or DM cache
  - Also called collision misses.

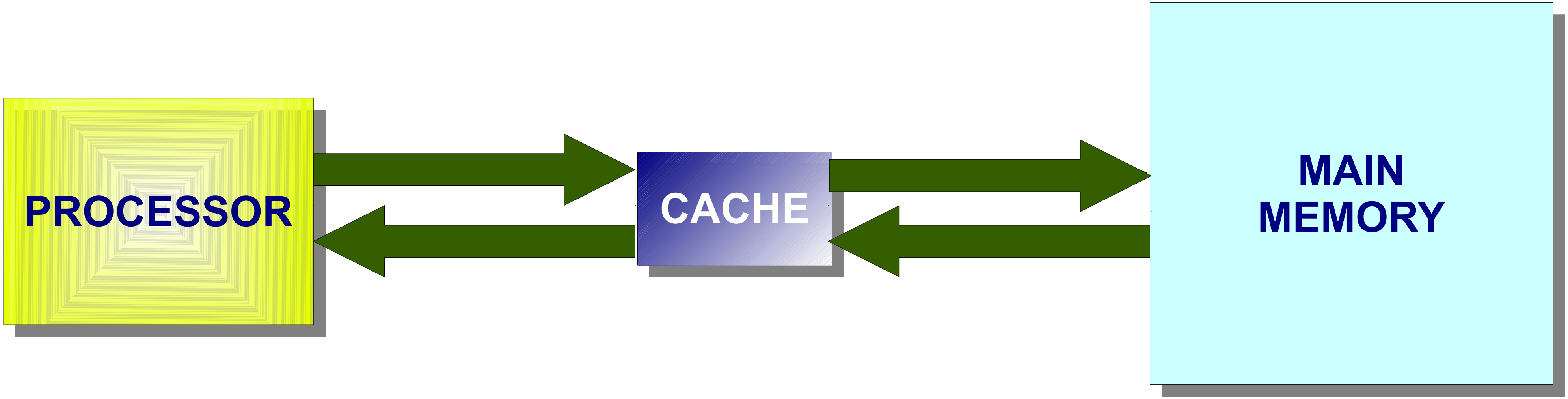
Events on a Cache Miss

- Stall the pipeline.
- Steps on an I-Cache miss:

1. Send the PC value to the memory.
2. Instruct main memory to perform a read and wait for the memory to complete its access.
3. Fill the cache entry: write the data from memory into the cache block, fill the tag field from the address, turn the valid bit on.
4. Restart the instruction execution at the first step, which will refetch the instruction, this time finding it in the cache.

Cache Access Time

- Hit Time
- Miss Penalty



$$Average\ Memory\ Access\ Time = Hit\ Time + \underbrace{Miss\ rate \times Miss\ penalty}_{\text{Stall Time}}$$

Processor Performance – No Cache

- 5GHz processor, cycle time = 0.2ns
- Memory access time = 100ns = 500 cycles
- Ignoring memory access, Clocks Per Instruction (CPI) = 1
- Assuming no memory data access:
  - $CPI_{no\text{-}cache} = 1 + \#stall\ cycles$
  - 1 + 500 = 501

P

Main Memory Access

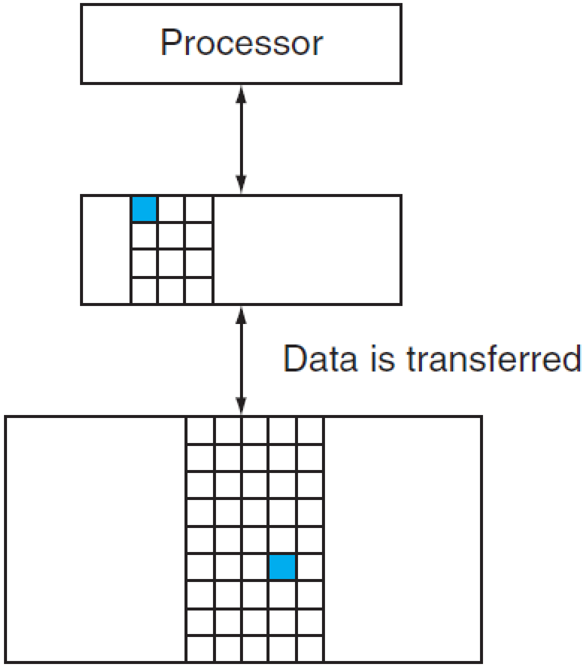
P

Main

Processor + Cache Performance

- Hit Rate = 0.95
- L1 Access Time = 0.2 ns = 1 cycle
- $\text{CPI}_{\text{with-cache}}$ = 1 + #stall cycles
- #stall cycles = ?
  - $stall\ cycles = Miss\ Rate \cdot Miss\ Penalty$
  - $stall\ cycles = 0.05 \cdot 500 = 25$
- $\text{CPI}_{\text{with-cache}}$ = 26
- Increase in performance = 501/26 = 19.3

Cache Terms



- cache hit – An access where the data is found in the cache.
- cache miss -- an access which isn’t
- hit time -- time to access the cache
- miss penalty -- time to move data from lower level to upper, then to cpu
- hit rate -- percentage of cache hits
- miss rate -- (1 – hit rate)
- cache block size or cache line size -- the amount of data that gets transferred on a cache miss.
- instruction cache -- cache that only holds instructions.
- data cache -- cache that only caches data.

Cache Performance

$$\text{CPU time} = (\text{CPU execution clock cycles} + \text{Memory-stall clock cycles}) \\ \times \text{Clock cycle time}$$

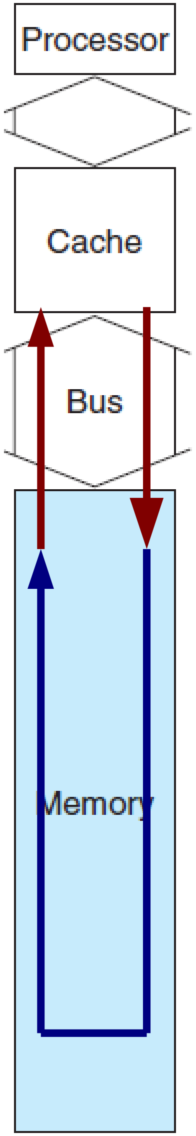
$$\text{Memory-stall clock cycles} = \frac{\text{Memory accesses}}{\text{Program}} \times \text{Miss rate} \times \text{Miss penalty}$$

Cache Performance

- Assume the miss rate of an instruction cache is 2% and the miss rate of the data cache is 4%. If a processor has a CPI of 2 without any memory stalls and the miss penalty is 100 cycles for all misses, determine how much faster a processor would run with a perfect cache that never missed. Assume the frequency of all loads and stores is 36%.

Average memory stall cycles for (a) Instruction Cache (b) Data Cache?

Main Memory Example

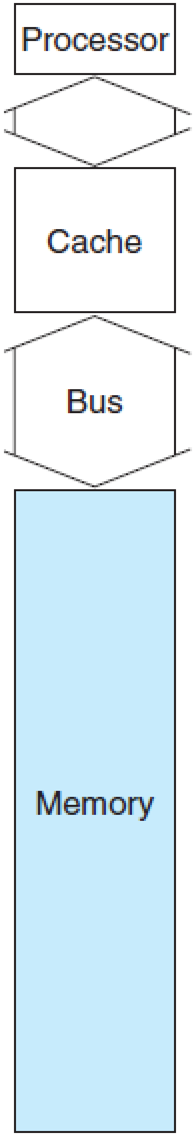


- Memory access times are given:
- 1 memory bus clock cycle to send the address
- 15 memory bus clock cycles for each DRAM access initiated
- 1 memory bus clock cycle to send a word of data
- What is the time taken to transfer 4 words from the DRAM? The DRAM bank is 1 word wide.

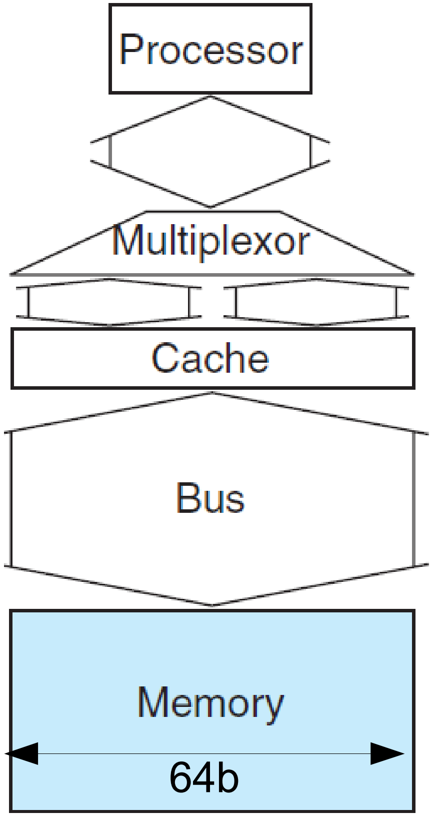
$$1+4\times 15+4\times 1=65\,cycles$$

$$\frac{4\times 4}{65}=\underbrace{0.25\,bytes\ per\ clock\ cycle}_{\textbf{Memory Bandwidth (bytes/sec)}}$$

Memory Organizations



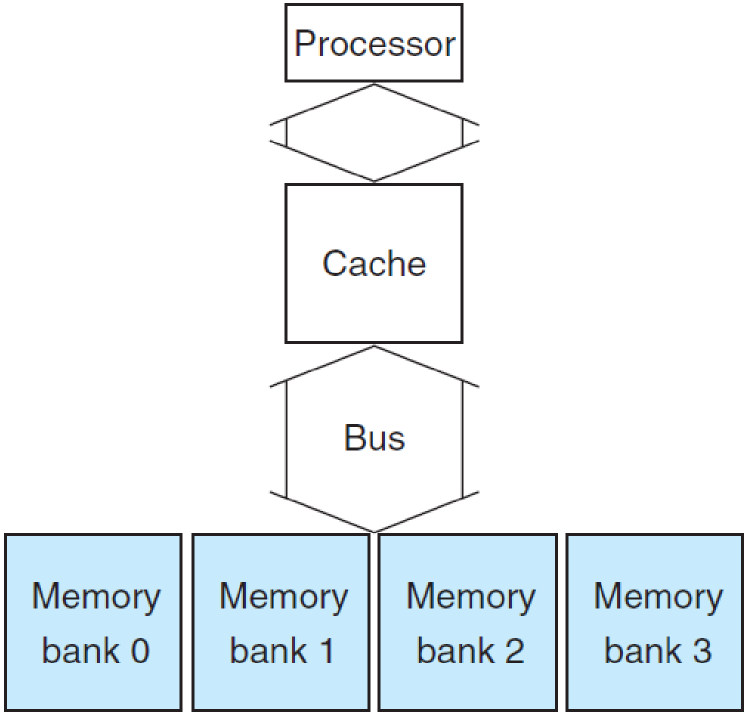
a. One-word-wide
memory organization



b. Wider memory organization

$$1+2\times15+2\times1=33\ cycles$$

$$\frac{2\times8}{33}=0.48\ bytes\ \ per\ clock\ cycle$$



c. Interleaved memory organization

$$1+1\times15+4\times1=20\ cycles$$

$$\frac{4\times4}{20}=0.8\ bytes\ \ per\ clock\ cycle$$