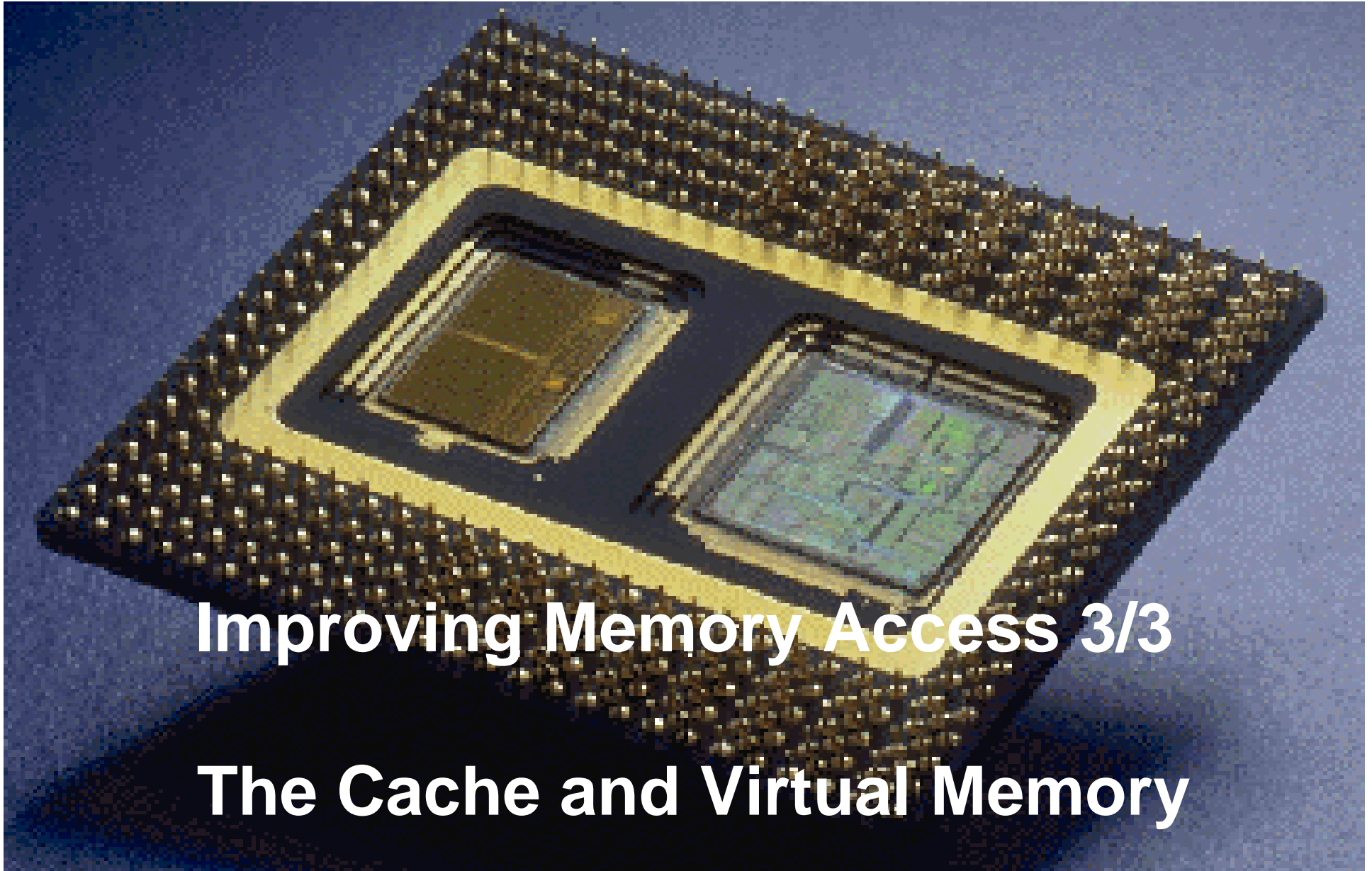


# EECS 322 Computer Architecture

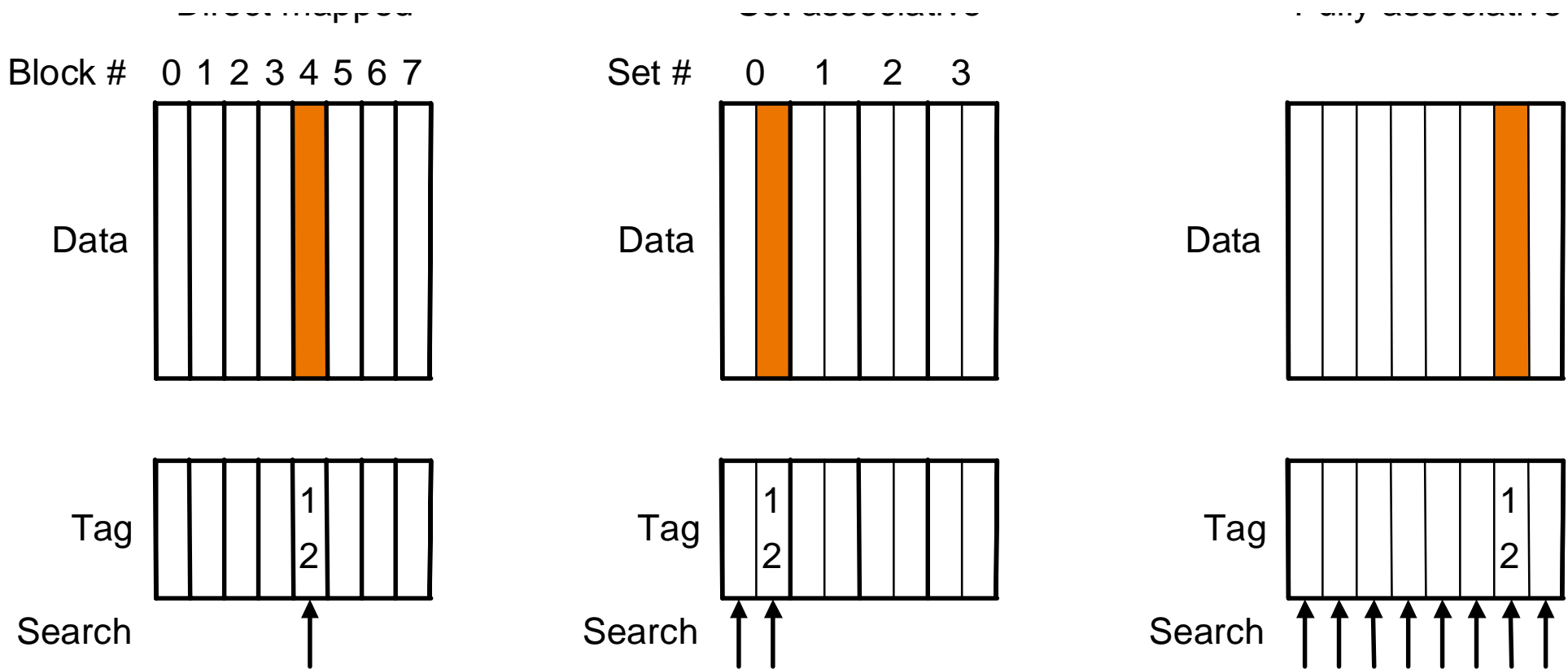


**Improving Memory Access 3/3**

**The Cache and Virtual Memory**

# Cache associativity

Figure 7.15



**Direct-mapped  
cache**

**2-way set  
associative  
cache**

**Fully  
associative  
cache**



# Direct Mapping Cache addresses



$$\text{Cache Index} = (\text{Block Address}) \% (\text{Cache Size})$$

$$\text{Cache Index} = \left\lfloor \frac{\text{Byte address}}{\text{Data Bytes per cache block}} \right\rfloor \% (\text{Cache Size})$$

$$\text{Block address} = \left\lfloor \frac{\text{Byte address}}{\text{Data Bytes per cache block}} \right\rfloor$$

$$\text{Cache Size in Blocks} = \frac{\text{Cache Size in Bytes}}{\text{Bytes per Cache Block}}$$

# N-Way Set-Associative Mapping Cache addresses



An *N*-way set associative cache consists of a number of **sets**, where each **set** consists on *N* blocks

$$\text{Cache Index} = \text{Set \#} = (\text{Block Address}) \% (\text{Cache Size})$$

$$\text{Cache Index} = \left\lfloor \frac{\text{Byte address}}{\text{Data Bytes per cache block}} \right\rfloor \% (\text{Cache Size})$$

$$\text{Block address} = \left\lfloor \frac{\text{Byte address}}{\text{Data Bytes per cache block}} \right\rfloor$$

$$\text{Cache Size in } N - \text{way Sets} = \frac{\text{Cache Size in Bytes}}{N \times \text{Data Bytes per Cache Block}}$$

# Direct Mapped Cache



Suppose we have the have the following byte addresses:

0, 32, 0, 24, 32

Given a direct mapped cache consisting of 16 data bytes

where **Data bytes per cache block = 4**

where **Cache size = 16 bytes / (4 bytes per block) = 4**

The cache block addresses are: 0, 8, 0, 6, 8

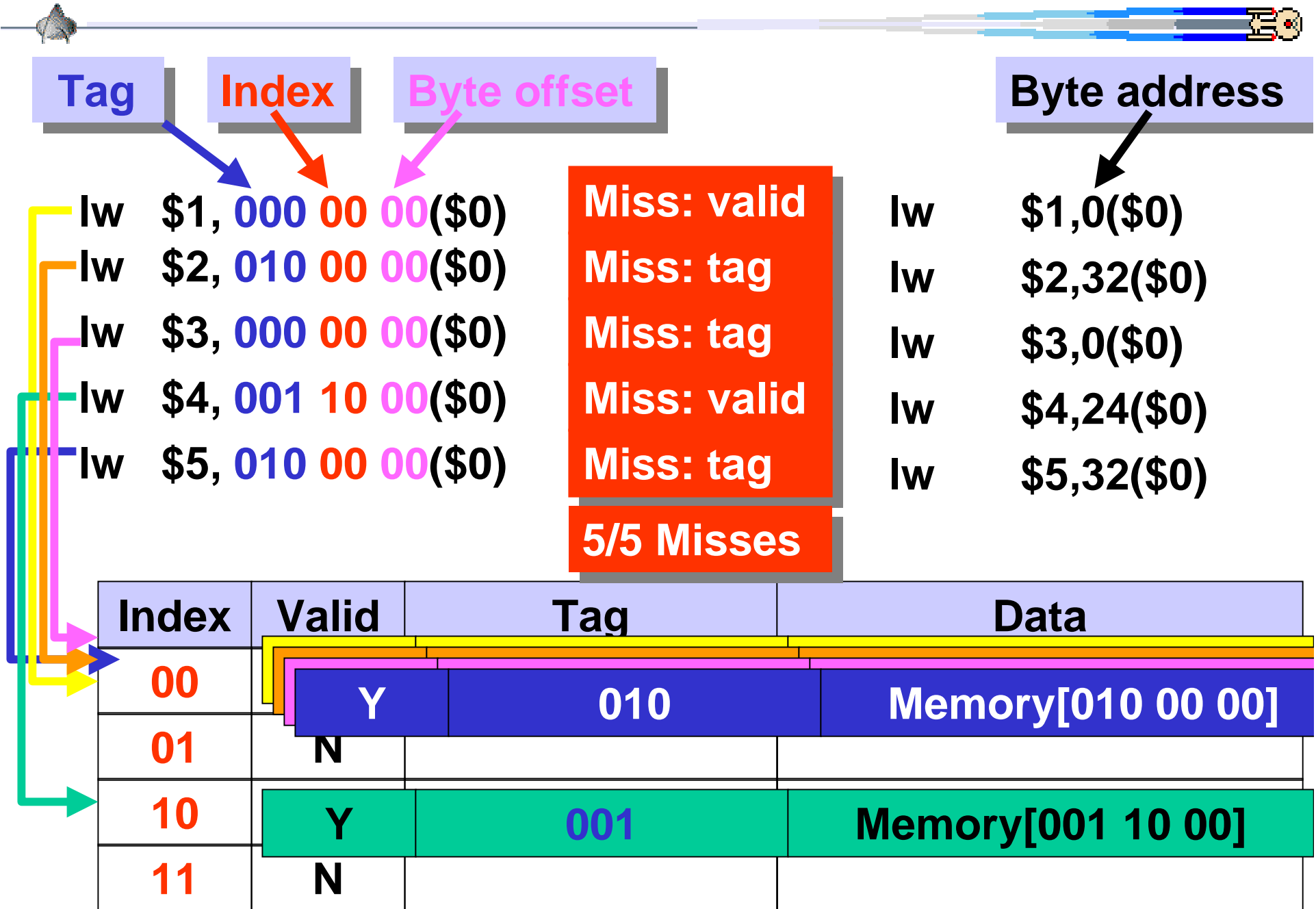
The cache index addresses are: 0, 0, 0, 2, 0

For example, 24,

**Block address = floor(byte address / data bytes per block)**  
**= floor(24/4)=6**

**Cache Index = 6 % Cache size = 6 % 4 = 2**

# Direct Mapped Cache: Example



# 2-Way Set associative cache



Again, suppose we have the have the following byte addresses:

0, 32, 0, 24, 32

Given a 2-way set associative cache of 16 data bytes

where **Data bytes per cache block = 4**

where **Cache size = 16 bytes / (2 × 4 bytes per block) = 2**

The 2-way cache block addresses are: 0, 8, 0, 6, 8

The 2-way cache index addresses are: 0, 0, 0, 0, 0

**Note: Direct cache index addresses were: 0, 0, 0, 2, 0**

For example, 24,

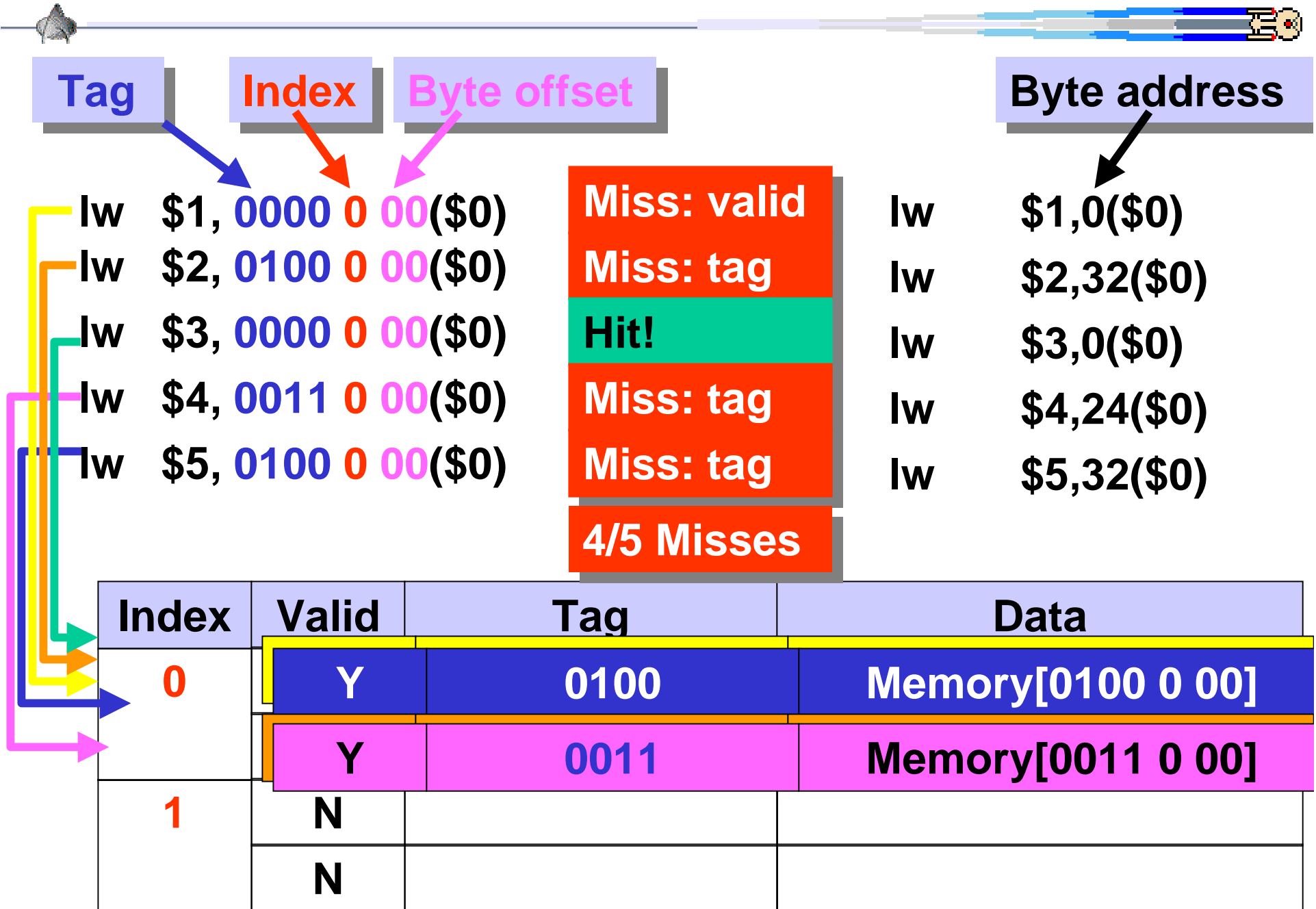
**Block address = floor(byte address / data bytes per block)**

**= floor(24/4) = 6**

**Cache Index = 6 % Cache size = 6 % 2 = 0**



# 2-Way Set Associative Cache: Example



# Fully associative cache



Again, suppose we have the have the following byte addresses:

0, 32, 0, 24, 32

Given a 2-way set associative cache of 16 data bytes

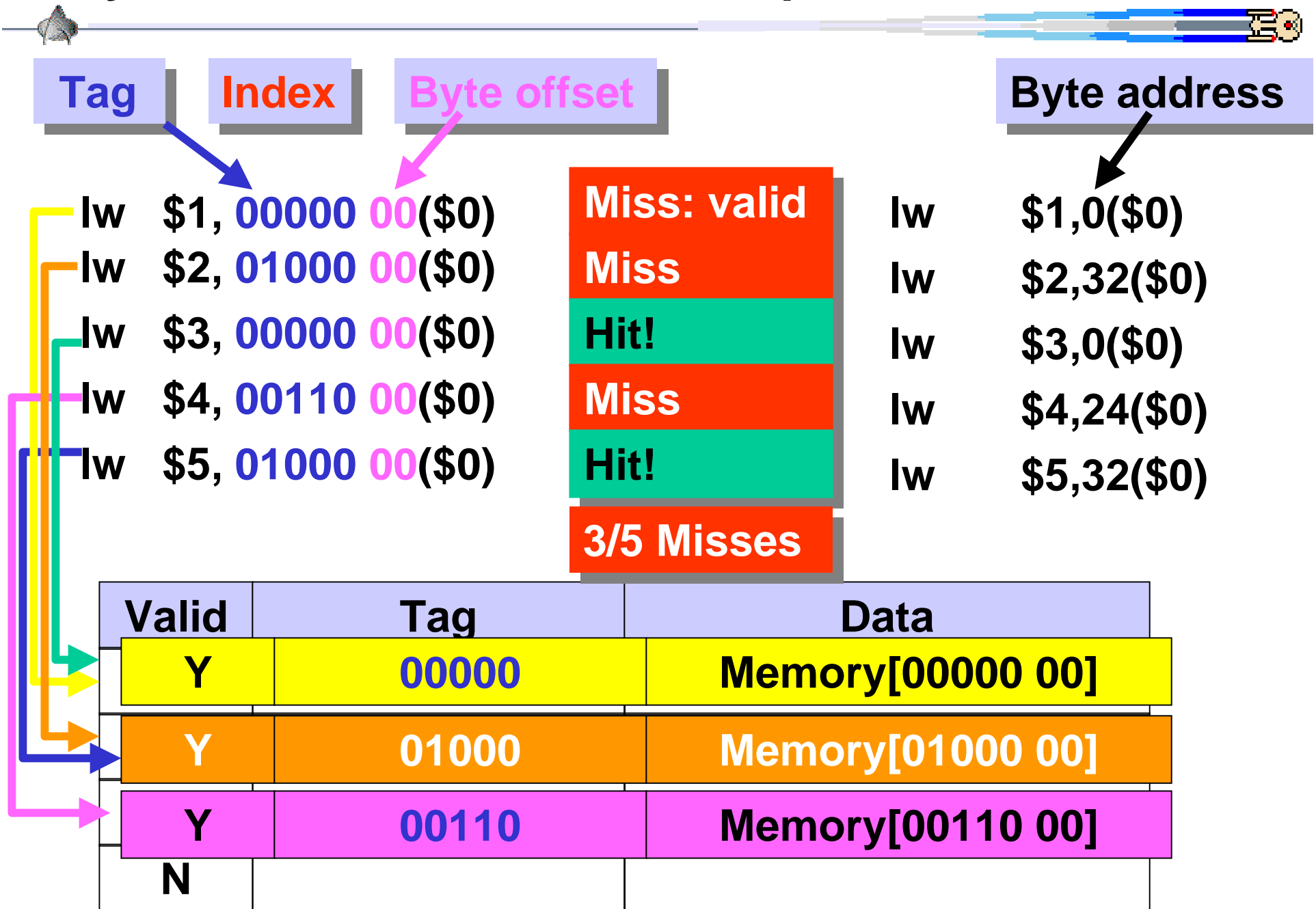
where **Data bytes per cache block = 4**

where **Cache size = 16 bytes/(4 bytes per block) = 4**

**The fully cache block addresses are: None!**

**The 2-way cache index addresses are: NONE!**

# Fully Associative Cache: Example



# Cache comparisons: Examples



Chip Area

Speed

**Direct Mapped cache:**

**5 out of 5 misses**

**2-Way Set associative cache:**

**4 out of 5 misses**

**Fully associative cache:**

**3 out of 5 misses**



# Bits in a Temporal Direct Mapped Cache

How many total bits are required for a cache with 64KB (=  $2^{16}$  KiloBytes) of data and one word (=4 bytes =32 bit) blocks assuming a 32 bit byte memory address?

$$\begin{aligned}\text{Cache index width} &= \log_2 \text{ words} \\ &= \log_2 2^{16}/4 = \log_2 2^{14} \text{ words} = 14 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Block address width} &= \langle \text{byte address width} \rangle - \log_2 \text{ word} \\ &= 32 - 2 = 30 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Tag size} &= \langle \text{block address width} \rangle - \langle \text{cache index width} \rangle \\ &= 30 - 14 = 16 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Cache block size} &= \langle \text{valid size} \rangle + \langle \text{tag size} \rangle + \langle \text{block data size} \rangle \\ &= 1 \text{ bit} + 16 \text{ bits} + 32 \text{ bits} = 49 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Total size} &= \langle \text{Cache word size} \rangle \times \langle \text{Cache block size} \rangle \\ &= 2^{14} \text{ words} \times 49 \text{ bits} = 784 \times 2^{10} = 784 \text{ Kbits} = 98 \text{ KB} \\ &= 98 \text{ KB}/64 \text{ KB} = 1.5 \text{ times overhead}\end{aligned}$$

# Bits in a Spatial Direct Mapped Cache

How many total bits are required for a cache with 64KB (=  $2^{16}$  KiloBytes) of data and 4 - one word (=  $4 \times 4 = 16$  bytes =  $4 \times 32 = 128$  bits) blocks assuming a 32 bit byte memory address?

$$\begin{aligned}\text{Cache index width} &= \log_2 \text{ words} \\ &= \log_2 2^{16}/16 = \log_2 2^{12} \text{ words} = 12 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Block address width} &= \langle \text{byte address width} \rangle - \log_2 \text{ blocksize} \\ &= 32 - 4 = 28 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Tag size} &= \langle \text{block address width} \rangle - \langle \text{cache index width} \rangle \\ &= 28 - 12 = 16 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Cache block size} &= \langle \text{valid size} \rangle + \langle \text{tag size} \rangle + \langle \text{block data size} \rangle \\ &= 1 \text{ bit} + 16 \text{ bits} + 128 \text{ bits} = 145 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Total size} &= \langle \text{Cache word size} \rangle \times \langle \text{Cache block size} \rangle \\ &= 2^{12} \text{ words} \times 145 \text{ bits} = 593920 \text{ bits} / (1024 \times 8) = 72.5 \text{ Kbyte} \\ &= 72.5 \text{ KB}/64 \text{ KB} = 1.13 \text{ times overhead}\end{aligned}$$

# Bits in a 2-Way Set Associative Cache

How many total bits are required for a cache with 64KB (=  $2^{16}$  KiloBytes) of data and 4 - one word (=  $4 \times 4 = 16$  bytes =  $4 \times 32 = 128$  bits) blocks assuming a 32 bit byte memory address?

$$\begin{aligned}\text{Cache index width} &= \log_2 \text{ words} \\ &= \log_2 2^{16} / (2 \times 16) = \log_2 2^{11} \text{ words} = 11 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Block address width} &= \langle \text{byte address width} \rangle - \log_2 \text{ blocksize} \\ &= 32 - 4 = 28 \text{ bits (same!)}\end{aligned}$$

$$\begin{aligned}\text{Tag size} &= \langle \text{block address width} \rangle - \langle \text{cache index width} \rangle \\ &= 28 - 11 = 17 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Cache Set size} &= \langle \text{valid size} \rangle + \langle \text{tag size} \rangle + \langle \text{block data size} \rangle \\ &= 1 \text{ bit} + 2 \times (17 \text{ bits} + 128 \text{ bits}) = 311 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Total size} &= \langle \text{Cache word size} \rangle \times \langle \text{Cache block size} \rangle \\ &= 2^{11} \text{ words} \times 311 \text{ bits} = 636928 \text{ bits} / (1024 \times 8) = 77.75 \text{ Kbyte} \\ &= 72.5 \text{ KB} / 64 \text{ KB} = 1.21 \text{ times overhead}\end{aligned}$$

# Bits in a 4-Way Set Associative Cache

How many total bits are required for a cache with 64KB (=  $2^{16}$  KiloBytes) of data and 4 - one word (=  $4 \times 4 = 16$  bytes =  $4 \times 32 = 128$  bits) blocks assuming a 32 bit byte memory address?

$$\begin{aligned}\text{Cache index width} &= \log_2 \text{ words} \\ &= \log_2 2^{16} / (4 \times 16) = \log_2 2^{10} \text{ words} = 10 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Block address width} &= \langle \text{byte address width} \rangle - \log_2 \text{ blocksize} \\ &= 32 - 4 = 28 \text{ bits (same!)}\end{aligned}$$

$$\begin{aligned}\text{Tag size} &= \langle \text{block address width} \rangle - \langle \text{cache index width} \rangle \\ &= 28 - 10 = 18 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Cache Set size} &= \langle \text{valid size} \rangle + \langle \text{tag size} \rangle + \langle \text{block data size} \rangle \\ &= 1 \text{ bit} + 4 \times (18 \text{ bits} + 128 \text{ bits}) = 585 \text{ bits}\end{aligned}$$

$$\begin{aligned}\text{Total size} &= \langle \text{Cache word size} \rangle \times \langle \text{Cache block size} \rangle \\ &= 2^{10} \text{ words} \times 585 \text{ bits} = 599040 \text{ bits} / (1024 \times 8) = 73.125 \text{ Kbyte} \\ &= 73.125 \text{ KB} / 64 \text{ KB} = 1.14 \text{ times overhead}\end{aligned}$$



# Bits in a Fully Associative Cache

How many total bits are required for a cache with 64KB (=  $2^{16}$  KiloBytes) of data and 4-one word (=  $4 \times 4 = 16$  bytes =  $4 \times 32 = 128$  bits) blocks assuming a 32 bit byte memory address?

Cache index width = 0

Block address width =  $\langle \text{byte address width} \rangle - \log_2 \text{blocksize}$   
=  $32 - 4 = 28$  bits (same!)

Tag size =  $\langle \text{block address width} \rangle - \langle \text{cache index width} \rangle$   
=  $28 - 0 = 28$  bits

Cache Set size =  $\langle \text{valid size} \rangle + \langle \text{tag size} \rangle + \langle \text{block data size} \rangle$   
= 1 bit + 28 bits + 128 bits = 157 bits

Total size =  $\langle \text{Cache word size} \rangle \times \langle \text{Cache block size} \rangle$   
=  $2^{12}$  words  $\times$  157 bits = 643072 bits /  $(1024 \times 8) = 78.5$  Kbyte  
= 78.5 KB/64 KB = 1.23 times overhead

# Bits in 64 KB Data Cache Summary



**Chip Area**

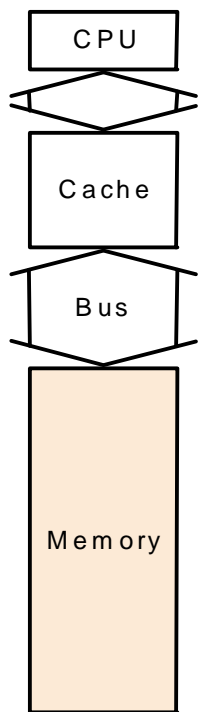
Cache Type	Index Size	Tag Size	Set Size	Over-head	Miss Rate
Temporal Direct Mapped	14 bits	16 bits	49 bits	1.5%	5.4%
Spatial Direct Mapped	12 bits	16 bits	145 bits	1.13%	1.9%
2 - Way Set Associative	11 bits	17 bits	311 bits	1.21%	1.5%
4 - Way Set Associative	10 bits	18 bits	585 bits	1.14%	1.5%
Fully Associative	0 bits	28 bits	157 bits	1.23%	



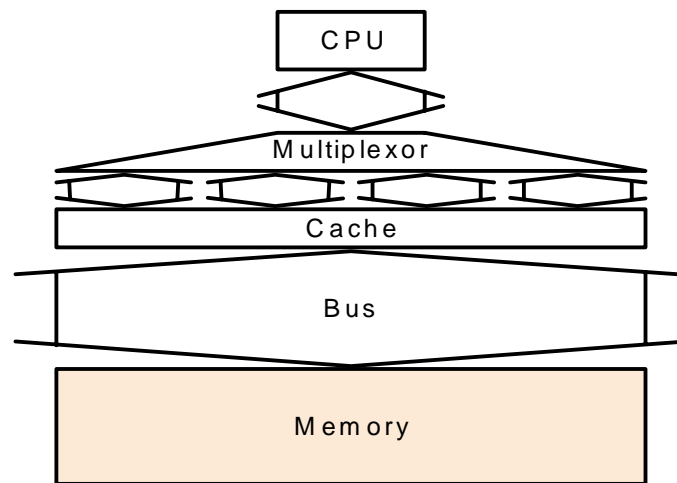
# Designing the Memory System

Figure 7.13

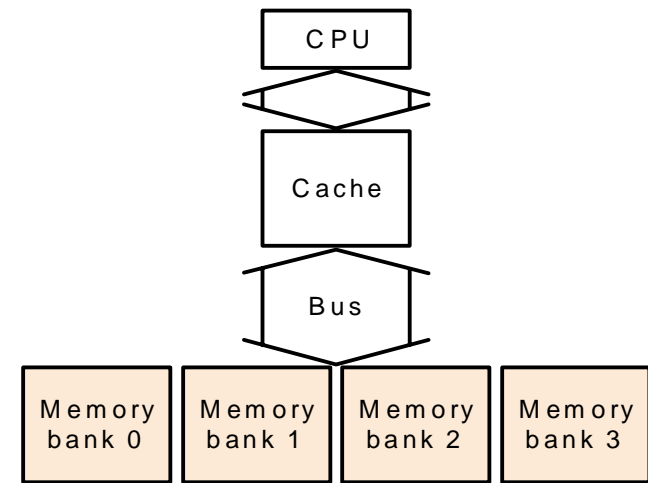
- Make reading multiple words easier by using banks of memory



a. One-word-wide memory organization



b. Wide memory organization



c. Interleaved memory organization

# Memory organizations

Figure 7.13



## One word wide memory organization

### Advantage

Easy to implement, low hardware overhead

### Disadvantage

Slow: **0.25 bytes/clock transfer rate**

## Interleave memory organization

### Advantage

Better: **0.80 bytes/clock transfer rate**

Banks are valuable on writes: independently

### Disadvantage

more complex bus hardware

## Wide memory organization

### Advantage

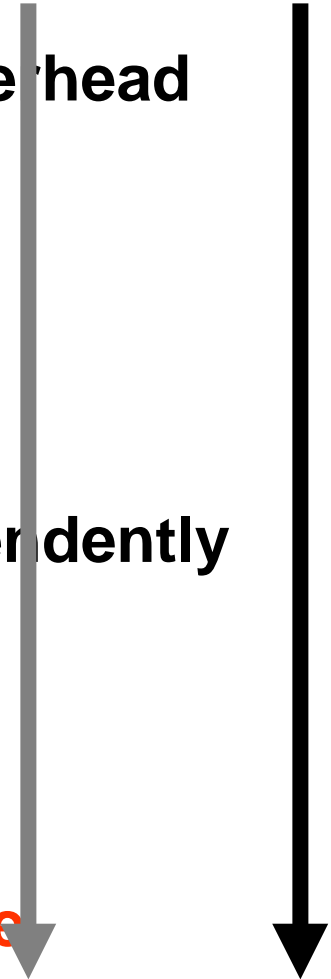
Fastest: **0.94 bytes/clock transfer rate**

### Disadvantage

Wider bus and increase in cache access time

Chip Area

Speed



# Decreasing miss penalty with multilevel caches Page 576



Suppose we have a processor with

<b>CPI</b>	<b>= 1.0</b>
<b>Clock Rate</b>	<b>= 500 Mhz = 2 ns</b>
<b>L1 Cache Miss rate</b>	<b>= 5%</b>
<b>DRAM</b>	<b>= 200 ns</b>

How much faster will the machine will be if we add a

<b>L2 Cache</b>	<b>= 20 ns (hit time = miss penalty)</b>
<b>L1 Cache Miss rate</b>	<b>= 2%</b>

$$\text{L to M Miss Penalty} = \frac{200 \text{ ns}}{2 \text{ ns per Clock Cycle}} = 100 \text{ Clock Cycles}$$

$$\text{L1 to L2 Miss Penalty} = \frac{20 \text{ ns}}{2 \text{ ns per Clock Cycle}} = 10 \text{ Clock Cycles}$$

# Decreasing miss penalty with multilevel caches Page 576



## The effective CPI with only L1 to M cache

$$\begin{aligned}\text{Total CPI} &= \text{Base CPI} + (\text{Memory Stall cycles per Instruction}) \\ &= 1.0 + 5\% \times 100\text{ns} = 6.0\end{aligned}$$

## The effective CPI with L1 and L2 caches

$$\begin{aligned}\text{Total CPI} &= \text{Base CPI} + (\text{L1 to L2 penalty}) + (\text{L2 to M penalty}) \\ &= 1.0 + (5\% \times 10\text{ns}) + (2\% \times 100\text{ns}) = 3.5\end{aligned}$$

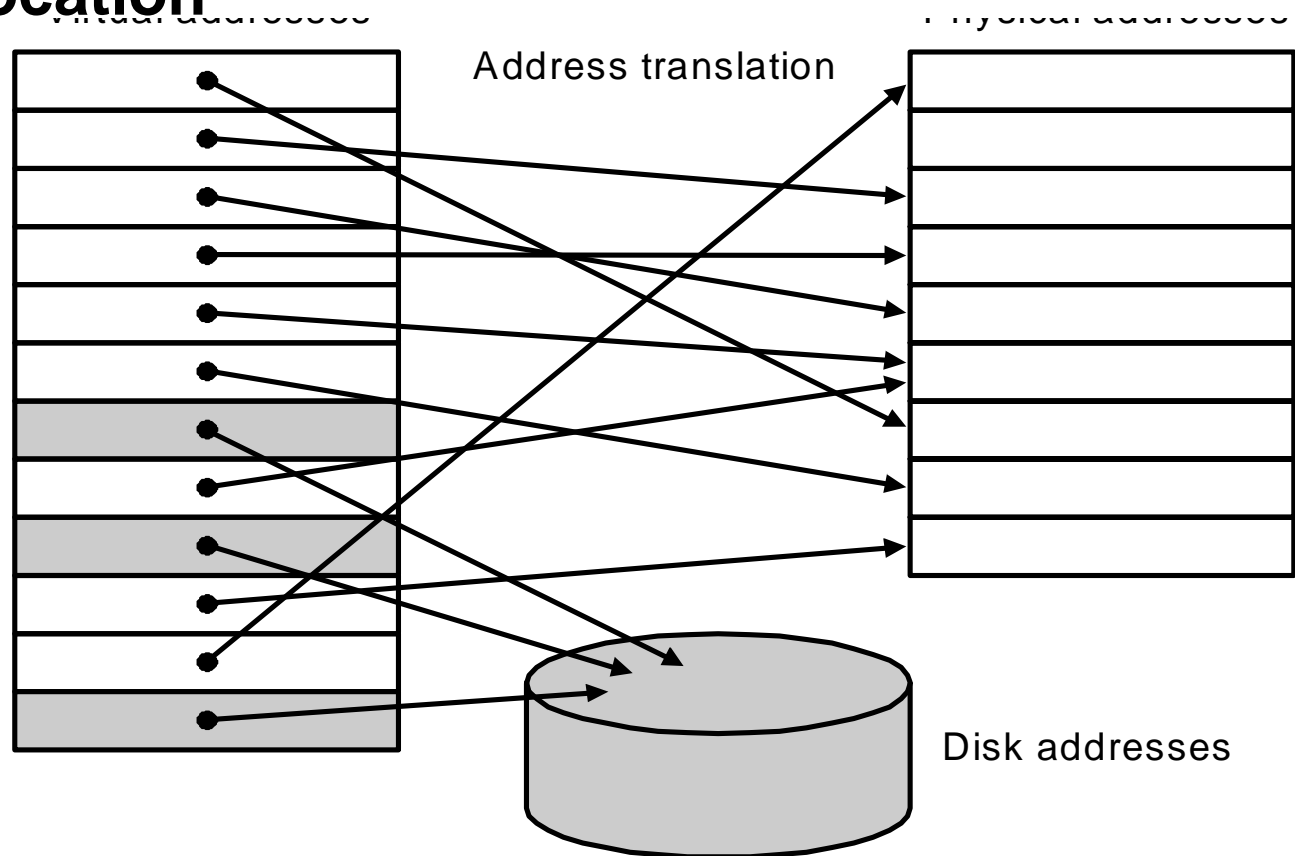
$$\text{CPU Speedup} = \frac{6.0}{3.5} = 1.7$$

# Virtual Memory

Figure 7.20

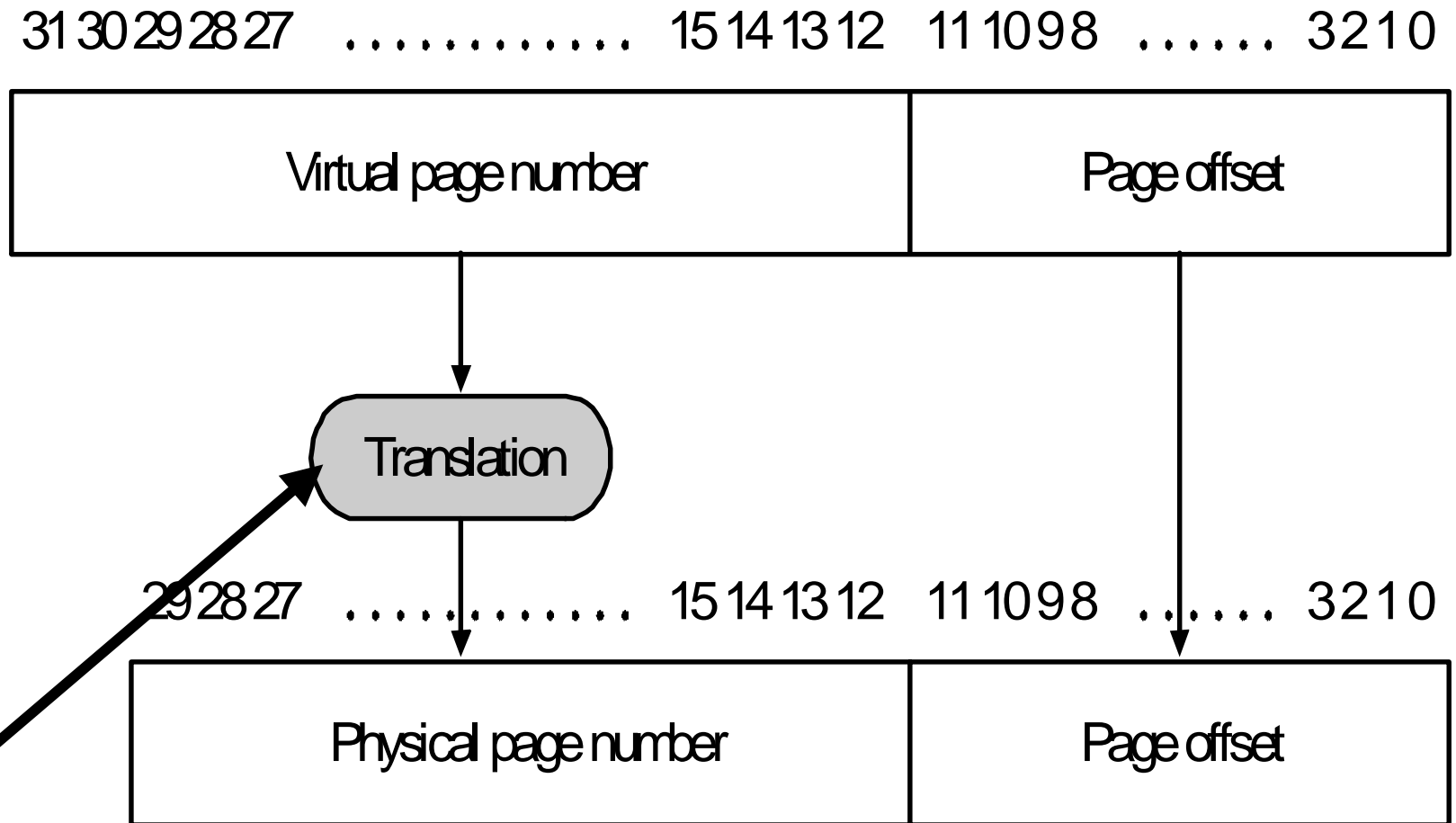


- **Main memory can act as a cache for the secondary storage (disk) Advantages:**
  - illusion of having more physical memory
  - **program relocation**
  - **protection**



# Pages: virtual memory blocks

Figure 7.21



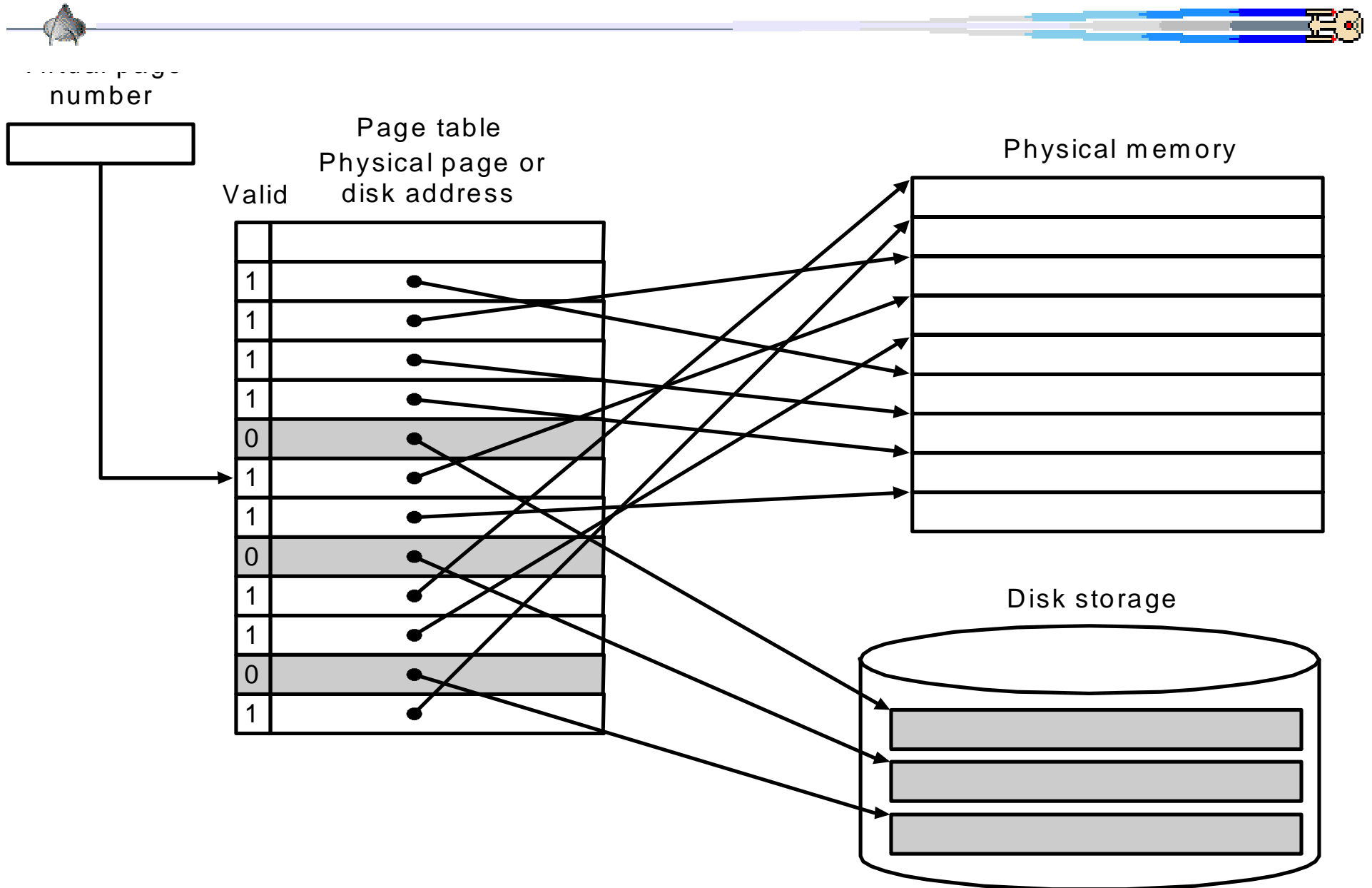
The automatic transmission of a car:  
Hardware/Software does the shifting

Physical address

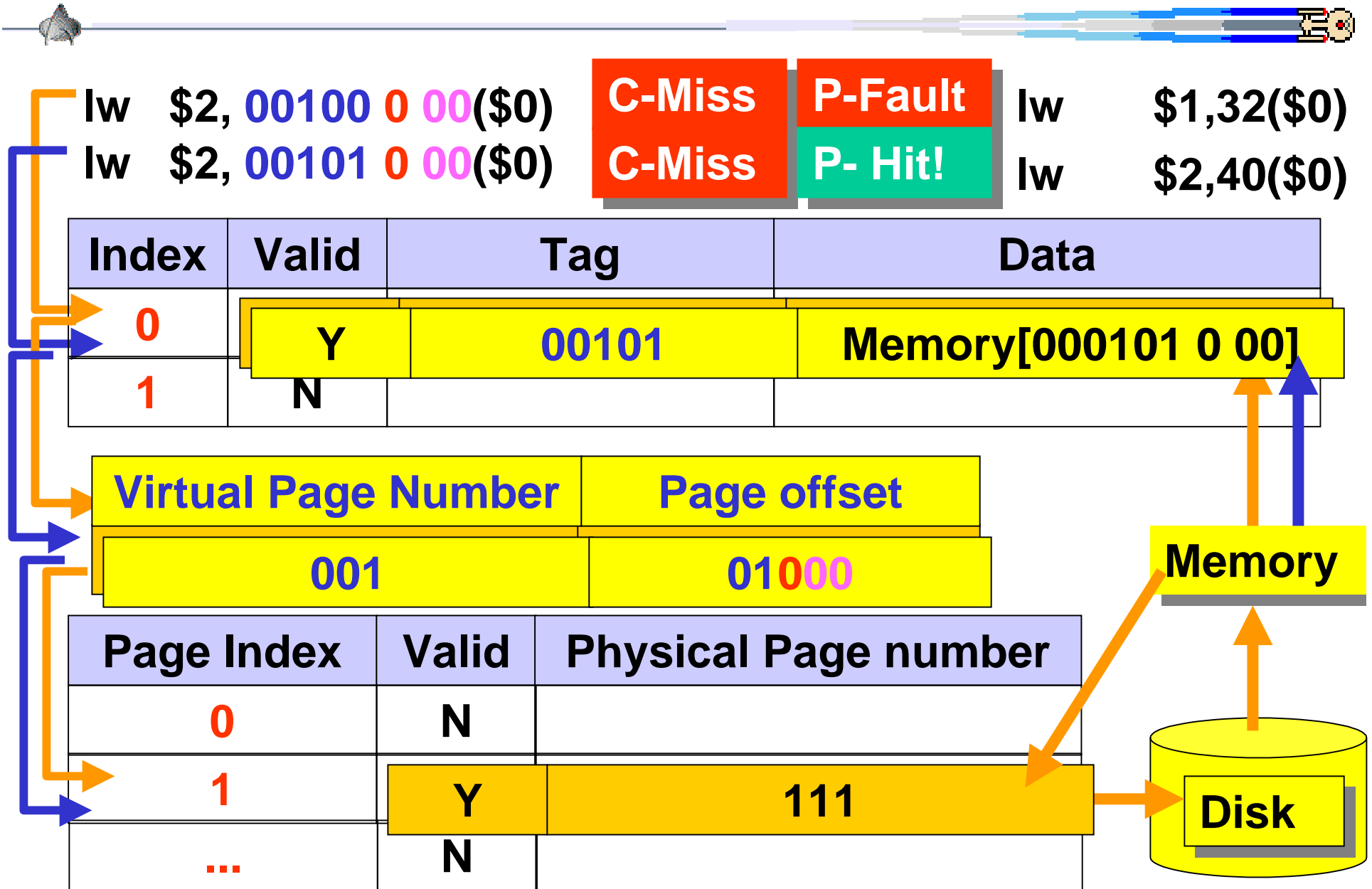


# Page Tables

Figure 7.23



# Virtual Example: 32 Byte pages



# Page Table Size



Suppose we have

32 bit virtual address ( $=2^{32}$  bytes)

4096 bytes per page ( $=2^{12}$  bytes)

4 bytes per page table entry ( $=2^2$  bytes)

What is the total page table size?

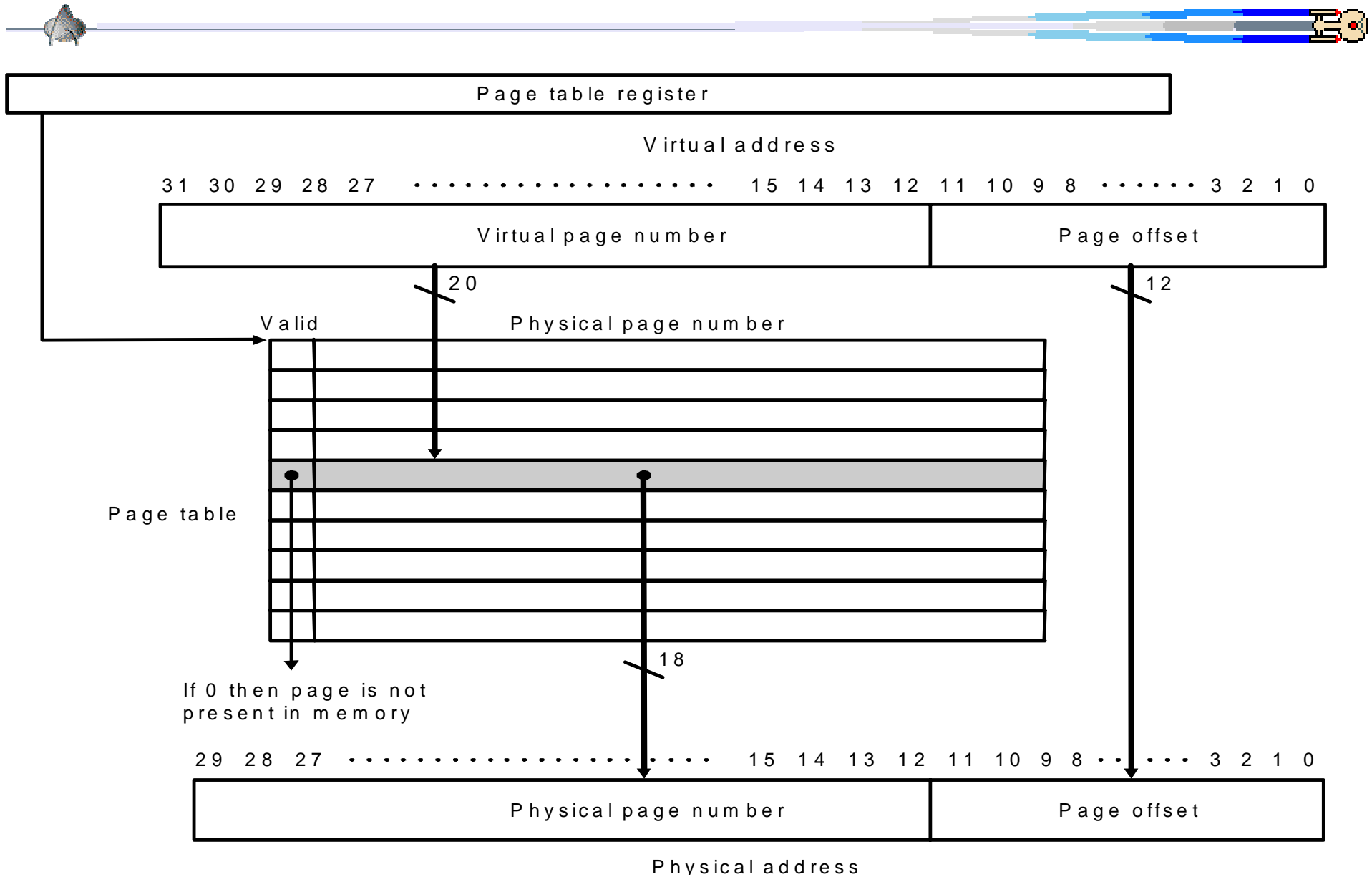
$$\text{Number of page table entries} = \frac{2^{32}}{2^{12}} = 2^{20} \text{ entries}$$

$$\text{Size of page table} = 2^{20} \text{ entries} \times 2^2 \text{ bytes} = 2^{22} \text{ bytes} = 4 \text{ MB}$$

**4 Megabytes just for page tables!! Too Big**

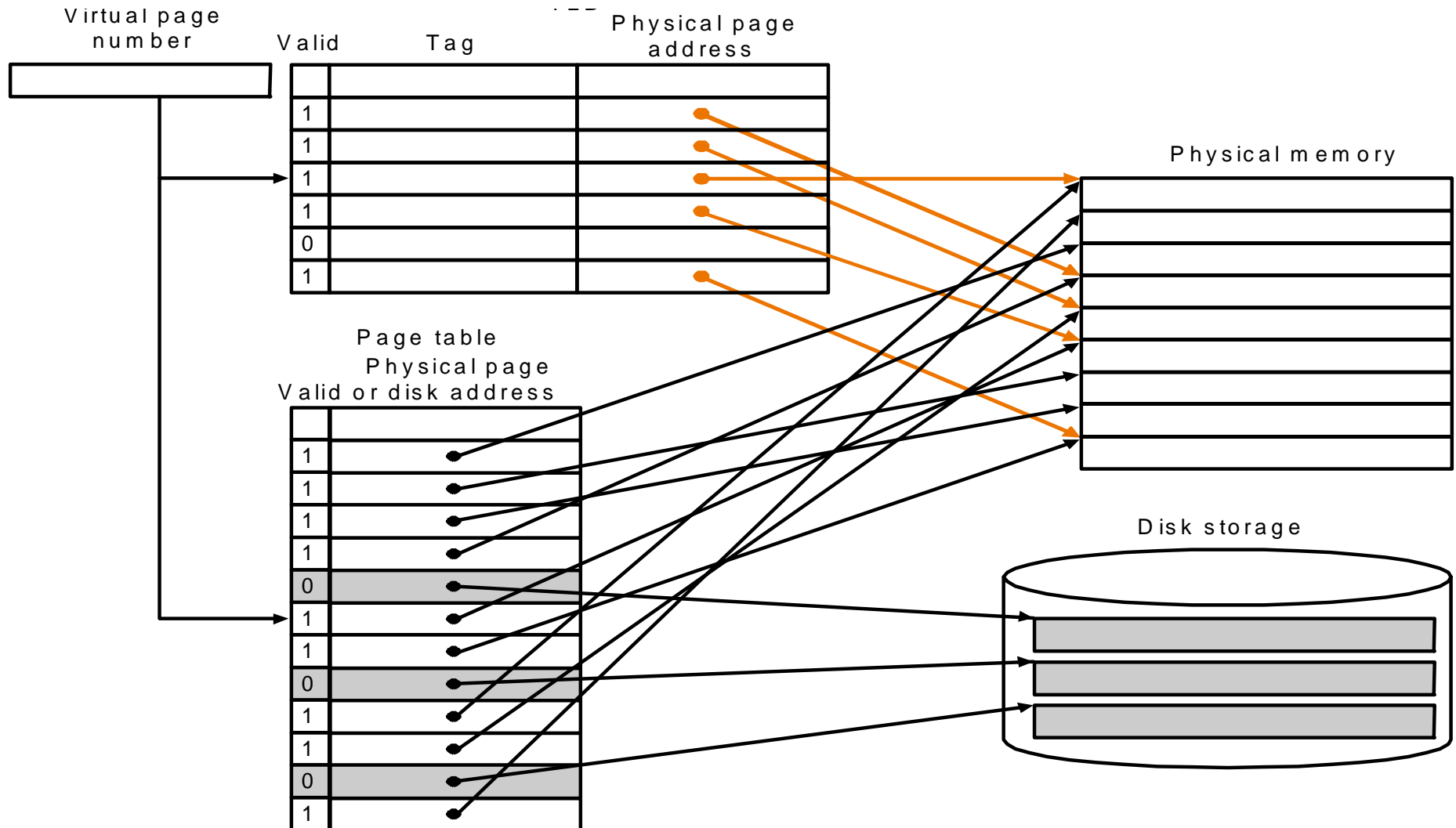
# Page Tables

Figure 7.22



# Making Address Translation Fast

- A cache for address translations: translation lookaside buffer



# TLBs and caches

