2010-06-03

2010年 前学期 TOKYO TECH

# 計算機アーキテクチャ 第一 (E)

## 7. メモリ2：半導体メモリシステム

吉瀬 謙二　計算工学専攻
kise_at_cs.titech.ac.jp
W641講義室　木曜日13:20 ー 14:50

## Acknowledgement

- Lecture slides for Computer Organization and Design, Third Edition, courtesy of **Professor Mary Jane Irwin**, Penn State University
- Lecture slides for Computer Organization and Design, third edition, Chapters 1-9, courtesy of **Professor Tod Amon,** Southern Utah University.

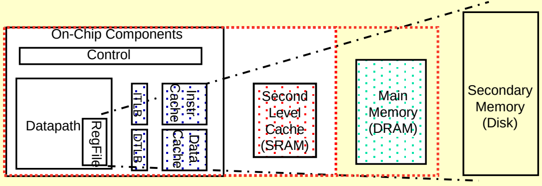
Adapted from *Computer Organization and Design*, Patterson & Hennessy, © 2005

## The Memory Hierarchy **Goal**

- **Fact**:
  **Large memories are slow** and
  **fast memories are small**
- How do we create a memory that gives the illusion of being large, cheap and fast ?
  - With **hierarchy**（階層）
  - With **parallelism**（並列性）

## A Typical Memory Hierarchy

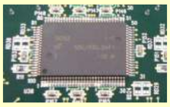
- By taking advantage of **the principle of locality**（局所性）
  - Present **much memory** in **the cheapest technology**
  - at **the speed of fastest technology**



| | | | | | |
|---|---|---|---|---|---|
| **Speed (%cycles):** | ½'s | 1's | 10's | 100's | 1,000's |
| **Size (bytes):** | 100's | K's | 10K's | M's | G's to T's |
| **Cost:** | highest | | | | lowest |

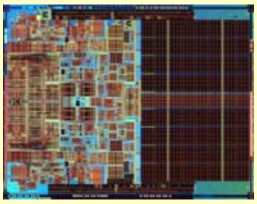
## DRAM (dynamic random access memory)

## SRAM (static random access memory)



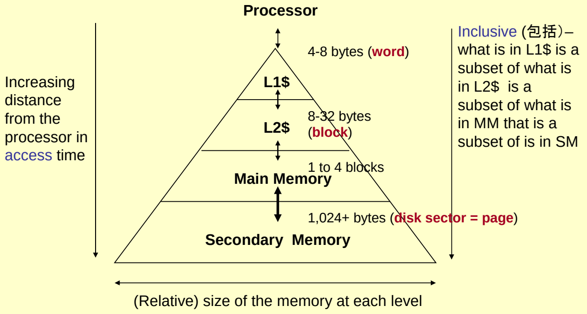
## Cache

- *Cache memory* consists of a small, fast memory that acts as a buffer for the DRAM memory.
- The nontechnical definition of *cache* is a safe place for hiding things.



Intel Core 2 Duo

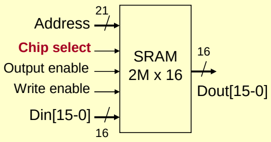
## Characteristics of the Memory Hierarchy



Processor
4-8 bytes (word)
L1$
8-32 bytes (block)
L2$
1 to 4 blocks
Main Memory
1,024+ bytes (disk sector = page)
Secondary Memory
Increasing distance from the processor in access time
Inclusive (包括)– what is in L1$ is a subset of what is in L2$ is a subset of what is in MM that is a subset of is in SM
(Relative) size of the memory at each level

## Memory Hierarchy Technologies

- Caches use SRAM for speed and technology compatibility
  - **Low density** (**6 transistor cells**), high power, expensive, fast
  - **Static**: content will last "forever" (until power turned off)

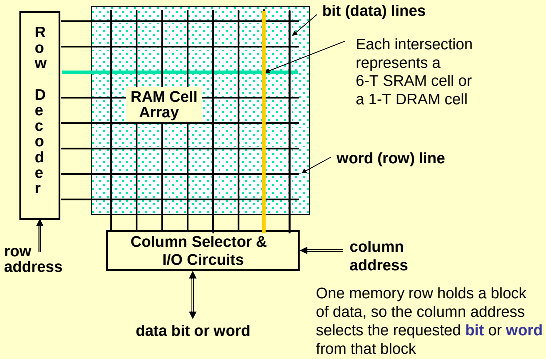


- Main Memory uses DRAM for size (density)
  - **High density** (**1 transistor cells**), low power, cheap, slow
  - **Dynamic**: needs to be "refreshed" regularly (~ every 8 ms)
    - 1% to 2% of the active cycles of the DRAM
  - Addresses divided into 2 halves (row and column)
    - **RAS** or Row Access Strobe triggering row decoder
    - **CAS** or Column Access Strobe triggering column selector

## Memory Performance Metrics

- **Latency**（レイテンシ，応答時間）: Time to access one word
  - **Cycle time**: time between requests
  - **Access time**: time between the request and when the data is available (or written)
  - Usually **cycle time** > **access time**
- **Bandwidth**（バンド幅，スループット）: How much data from the memory can be supplied to the processor per unit time
  - width of the data channel * the rate at which it can be used

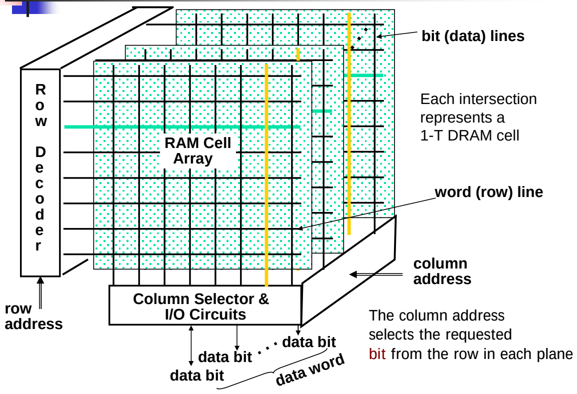
## Classical RAM Organization (~Square)



Each intersection represents a 6-T SRAM cell or a 1-T DRAM cell

One memory row holds a block of data, so the column address selects the requested **bit** or **word** from that block

## Classical DRAM Organization (~Square Planes)

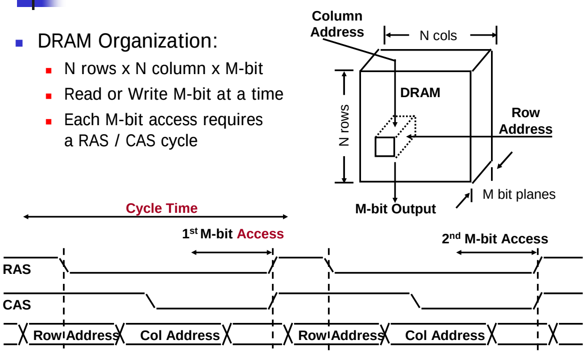


Each intersection represents a 1-T DRAM cell

The column address selects the requested bit from the row in each plane

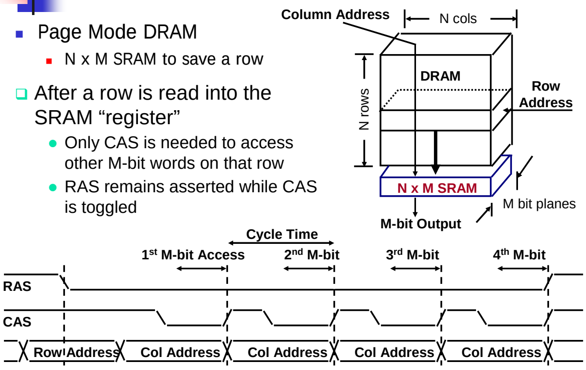
## Classical **DRAM Operation**

- DRAM Organization:
  - N rows x N column x M-bit
  - Read or Write M-bit at a time
  - Each M-bit access requires a RAS / CAS cycle



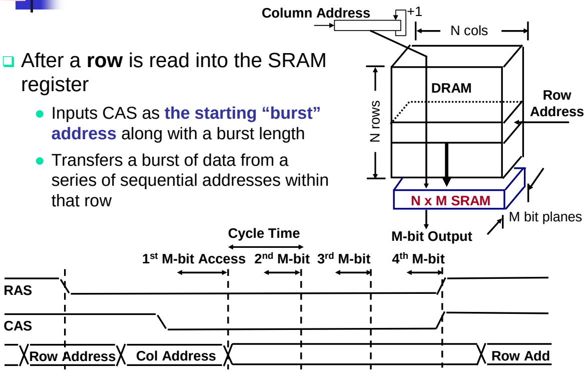
## **Page Mode** DRAM Operation

- Page Mode DRAM
  - N x M SRAM to save a row
- ❑ After a row is read into the SRAM "register"
  - Only CAS is needed to access other M-bit words on that row
  - RAS remains asserted while CAS is toggled



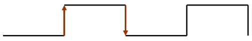
## **Synchronous DRAM** (**SDRAM**) Operation

- ❑ After a **row** is read into the SRAM register
  - Inputs CAS as **the starting "burst" address** along with a burst length
  - Transfers a burst of data from a series of sequential addresses within that row



## Other DRAM Architectures

- Double Data Rate SDRAMs – **DDR-SDRAMs** (and DDR-SRAMs)
  - Double data rate because they transfer data on both the rising and falling edge of the clock
  - Are the most widely used form of SDRAMs
- **DDR2-SDRAMs**
- **DDR3-SDRAMs**



## DRAM Memory Latency & Bandwidth Milestones

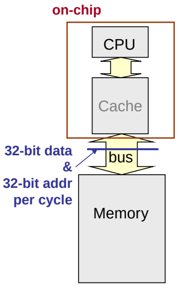
| | DRAM | Page DRAM | FastPage DRAM | FastPage DRAM | Synch DRAM | DDR SDRAM |
|---|---|---|---|---|---|---|
| Module Width | 16b | 16b | 32b | 64b | 64b | 64b |
| Year | 1980 | 1983 | 1986 | 1993 | 1997 | 2000 |
| Mb/chip | 0.06 | 0.25 | 1 | 16 | 64 | 256 |
| Die size (mm$^2$) | 35 | 45 | 70 | 130 | 170 | 204 |
| Pins/chip | 16 | 16 | 18 | 20 | 54 | 66 |
| **BWidth (MB/s)** | **13** | **40** | **160** | **267** | **640** | **1600** |
| **Latency (nsec)** | **225** | **170** | **125** | **75** | **62** | **52** |

Patterson, CACM Vol 47, #10, 2004

- In the time that the memory to processor bandwidth doubles the memory latency improves by a factor of only 1.2 to 1.4
- To deliver such high bandwidth, the internal DRAM has to be organized as **interleaved memory banks**

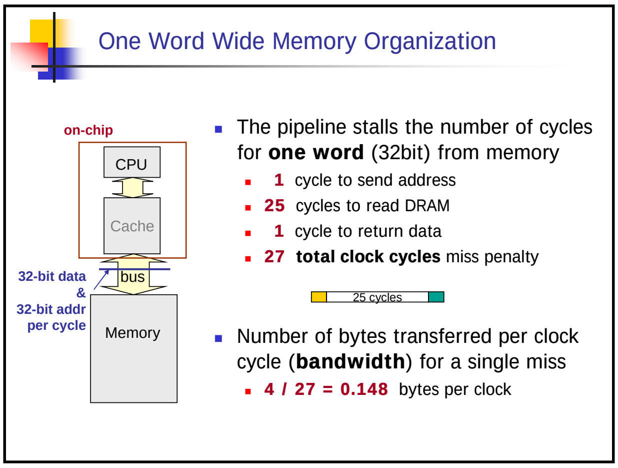
## Memory Systems that Support Caches

- The off-chip interconnect and memory architecture can affect overall system performance **in dramatic ways**



One word wide organization (one word wide bus and one word wide memory)

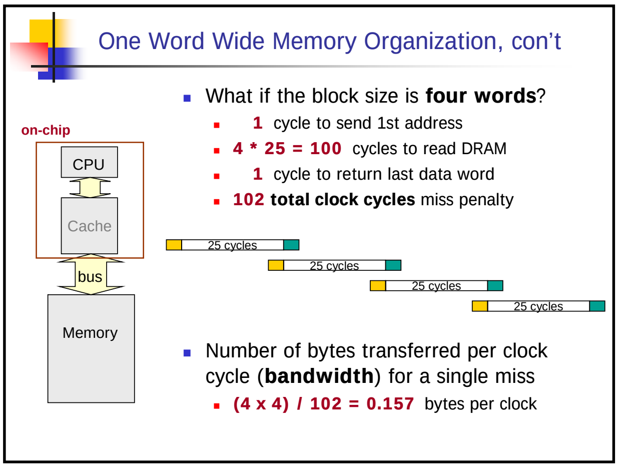
- ❑ **Assume**
  1. **1** clock cycle to send the address
  2. **25** clock cycles for DRAM **cycle time**, **8** clock cycles **access time**
  3. **1** clock cycle to return a word of data
- ❑ Memory-Bus to Cache **bandwidth**
  - **number of bytes transferred from memory to cache per clock cycle**



## One Word Wide Memory Organization

on-chip
CPU
Cache
bus
32-bit data & 32-bit addr per cycle
Memory
25 cycles

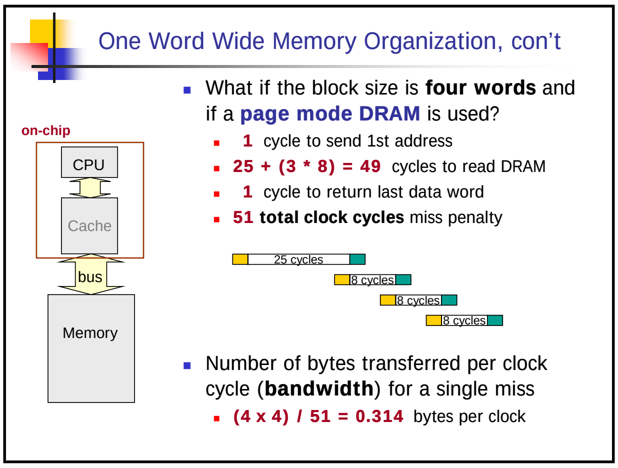
- The pipeline stalls the number of cycles for **one word** (32bit) from memory
  - **1** cycle to send address
  - **25** cycles to read DRAM
  - **1** cycle to return data
  - **27 total clock cycles** miss penalty
- Number of bytes transferred per clock cycle (**bandwidth**) for a single miss
  - **4 / 27 = 0.148** bytes per clock



## One Word Wide Memory Organization, con't

on-chip
CPU
Cache
bus
Memory
25 cycles
25 cycles
25 cycles
25 cycles

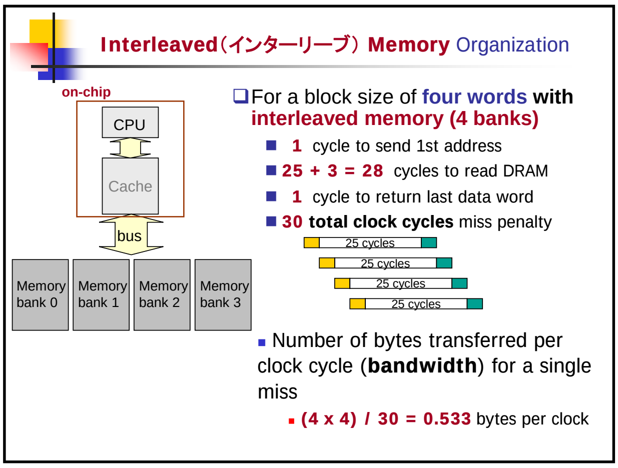
- What if the block size is **four words**?
  - **1** cycle to send 1st address
  - **4 * 25 = 100** cycles to read DRAM
  - **1** cycle to return last data word
  - **102 total clock cycles** miss penalty
- Number of bytes transferred per clock cycle (**bandwidth**) for a single miss
  - **(4 x 4) / 102 = 0.157** bytes per clock



## One Word Wide Memory Organization, con't

on-chip
CPU
Cache
bus
Memory
25 cycles
8 cycles
8 cycles
8 cycles

- What if the block size is **four words** and if a **page mode DRAM** is used?
  - **1** cycle to send 1st address
  - **25 + (3 * 8) = 49** cycles to read DRAM
  - **1** cycle to return last data word
  - **51 total clock cycles** miss penalty
- Number of bytes transferred per clock cycle (**bandwidth**) for a single miss
  - **(4 x 4) / 51 = 0.314** bytes per clock



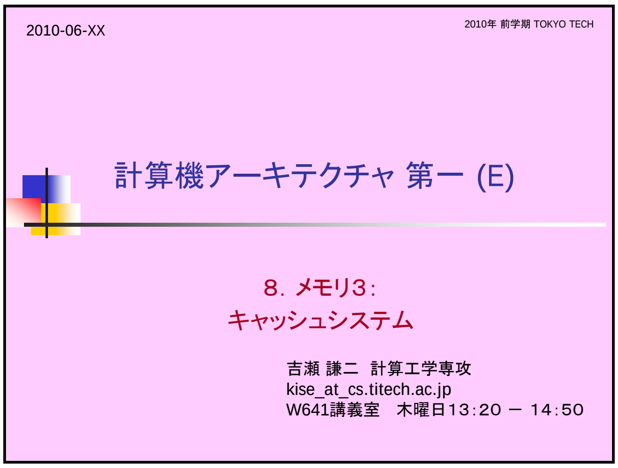
## **Interleaved(インターリーブ) Memory** Organization

on-chip
CPU
Cache
bus
Memory bank 0
Memory bank 1
Memory bank 2
Memory bank 3
25 cycles
25 cycles
25 cycles
25 cycles

❑ For a block size of **four words with interleaved memory (4 banks)**

- **1** cycle to send 1st address
- **25 + 3 = 28** cycles to read DRAM
- **1** cycle to return last data word
- **30 total clock cycles** miss penalty

- Number of bytes transferred per clock cycle (**bandwidth**) for a single miss
  - **(4 x 4) / 30 = 0.533** bytes per clock



2010-06-XX

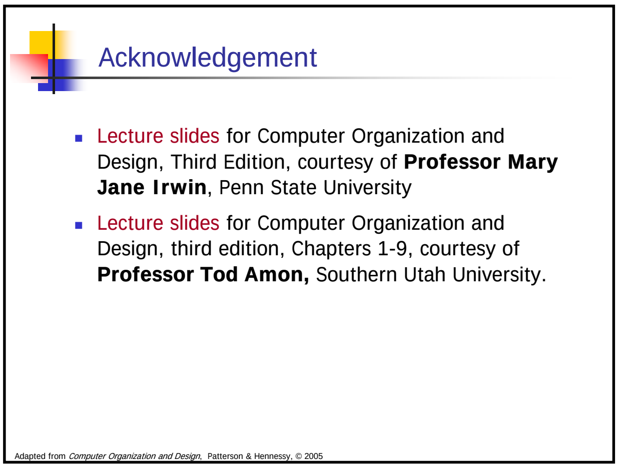
2010年 前学期 TOKYO TECH

# 計算機アーキテクチャ 第一 (E)

8. メモリ3:
キャッシュシステム

吉瀬 謙二 計算工学専攻
kise_at_cs.titech.ac.jp

W641講義室 木曜日13:20 − 14:50



## Acknowledgement

- Lecture slides for Computer Organization and Design, Third Edition, courtesy of **Professor Mary Jane Irwin**, Penn State University
- Lecture slides for Computer Organization and Design, third edition, Chapters 1-9, courtesy of **Professor Tod Amon,** Southern Utah University.

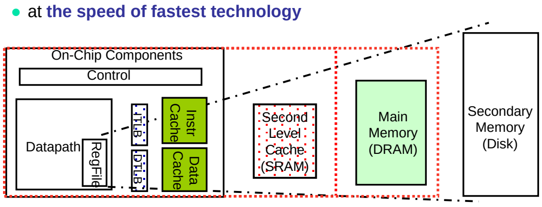
Adapted from *Computer Organization and Design*, Patterson & Hennessy, © 2005

## The Memory Hierarchy **Goal**

- **Fact**:
  **Large memories are slow** and
  **fast memories are small**
- How do we create a memory that gives the illusion of being large, cheap and fast ?
  - With **hierarchy**（階層）
  - With **parallelism**（並列性）

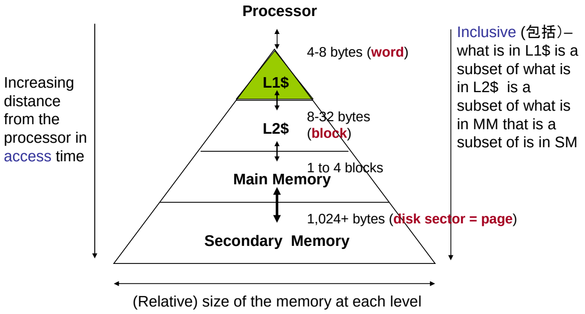
## A Typical Memory Hierarchy

- By taking advantage of **the principle of locality**（局所性）
  - Present **much memory** in **the cheapest technology**
  - at **the speed of fastest technology**



| | | | | | |
|---|---|---|---|---|---|
| **Speed (%cycles):** | ½'s | 1's | 10's | 100's | 1,000's |
| **Size (bytes):** | 100's | K's | 10K's | M's | G's to T's |
| **Cost:** | highest | | | | lowest |

## **Characteristics** of the **Memory Hierarchy**


Processor

4-8 bytes (**word**)

L1$

8-32 bytes (**block**)

L2$

1 to 4 blocks

Main Memory

1,024+ bytes (**disk sector = page**)

Secondary Memory

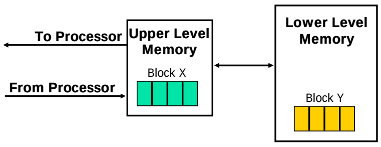
Increasing distance from the processor in access time

Inclusive (包括)– what is in L1$ is a subset of what is in L2$ is a subset of what is in MM that is a subset of is in SM

(Relative) size of the memory at each level

## The Memory Hierarchy: Why Does it Work?

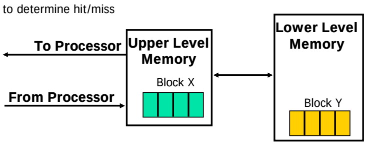
- **Temporal Locality** (時間的局所性, Locality in Time):
  ⇒ Keep **most recently accessed** data items closer to the processor
- **Spatial Locality** (空間的局所性, Locality in Space):
  ⇒ Move blocks consisting of **contiguous words** to the upper levels



28

## The Memory Hierarchy: **Terminology**

- **Hit**: data is in some block in the upper level (Block X)
  - **Hit Rate**: the fraction of memory accesses found in the upper level
  - **Hit Time**: Time to access the upper level which consists of
    RAM access time + Time to determine hit/miss



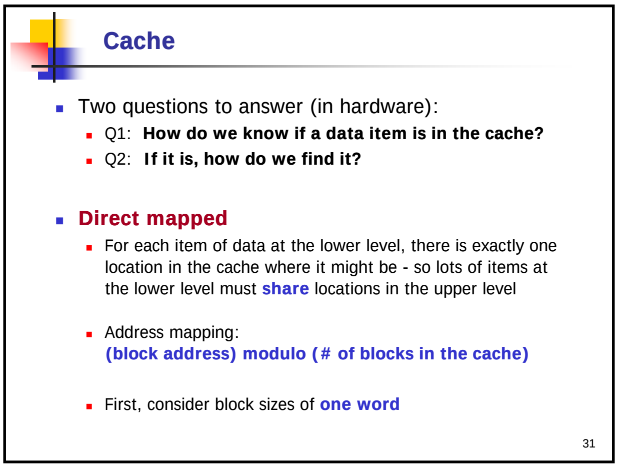
- **Miss**: data is not in the upper level so needs to be retrieve from a block in the lower level (Block Y)
  - **Miss Rate** = 1 - (Hit Rate)
  - **Miss Penalty**: Time to replace a block in the upper level
    + Time to deliver the block the processor
  - Hit Time << Miss Penalty

29

## How is the Hierarchy Managed?

- registers ↔ memory
  - by compiler (programmer?)
- cache ↔ main memory
  - by the cache controller hardware
- main memory ↔ disks
  - by the operating system (**virtual memory**)
    - virtual to physical address mapping assisted by the hardware (TLB, Translation Look-aside Buffer)
  - by the programmer (**files**)

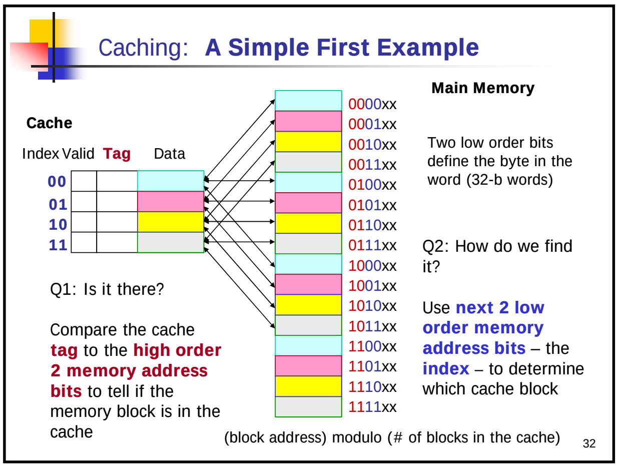
30



## Cache

- Two questions to answer (in hardware):
  - Q1: **How do we know if a data item is in the cache?**
  - Q2: **If it is, how do we find it?**
- **Direct mapped**
  - For each item of data at the lower level, there is exactly one location in the cache where it might be - so lots of items at the lower level must **share** locations in the upper level
  - Address mapping:<br>**(block address) modulo (# of blocks in the cache)**
  - First, consider block sizes of **one word**

31



## Caching: **A Simple First Example**

Two low order bits define the byte in the word (32-b words)

Q1: Is it there?

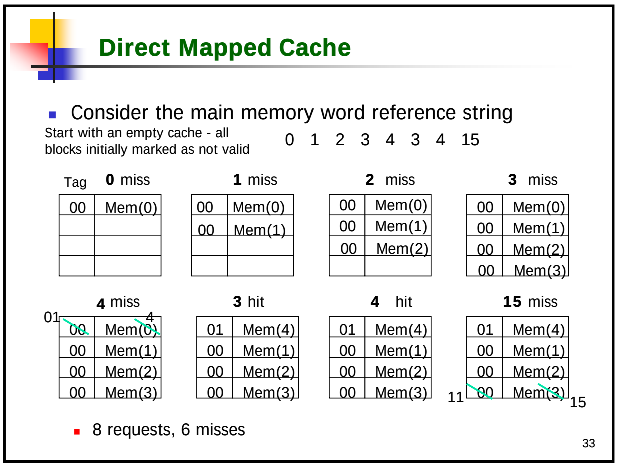
Compare the cache **tag** to the **high order 2 memory address bits** to tell if the memory block is in the cache

Q2: How do we find it?

Use **next 2 low order memory address bits** – the **index** – to determine which cache block

(block address) modulo (# of blocks in the cache)

32



## Direct Mapped Cache

- Consider the main memory word reference string

Start with an empty cache - all blocks initially marked as not valid

0 1 2 3 4 3 4 15

**0** miss

| Tag | |
|---|---|
| 00 | Mem(0) |
| | |
| | |
| | |

**1** miss

| | |
|---|---|
| 00 | Mem(0) |
| 00 | Mem(1) |
| | |
| | |

**2** miss

| | |
|---|---|
| 00 | Mem(0) |
| 00 | Mem(1) |
| 00 | Mem(2) |
| | |

**3** miss

| | |
|---|---|
| 00 | Mem(0) |
| 00 | Mem(1) |
| 00 | Mem(2) |
| 00 | Mem(3) |

**4** miss

| | |
|---|---|
| 01 ~~00~~ | 4 ~~Mem(0)~~ |
| 00 | Mem(1) |
| 00 | Mem(2) |
| 00 | Mem(3) |

**3** hit

| | |
|---|---|
| 01 | Mem(4) |
| 00 | Mem(1) |
| 00 | Mem(2) |
| 00 | Mem(3) |

**4** hit

| | |
|---|---|
| 01 | Mem(4) |
| 00 | Mem(1) |
| 00 | Mem(2) |
| 00 | Mem(3) |

**15** miss

| | |
|---|---|
| 01 | Mem(4) |
| 00 | Mem(1) |
| 00 | Mem(2) |
| 11 ~~00~~ | ~~Mem(3)~~ 15 |

- 8 requests, 6 misses

33