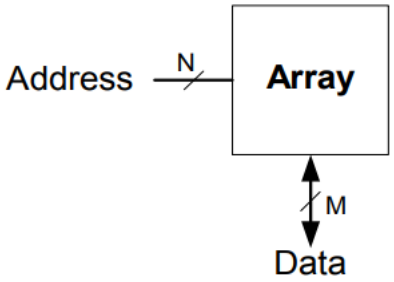
Memory

1 Memory

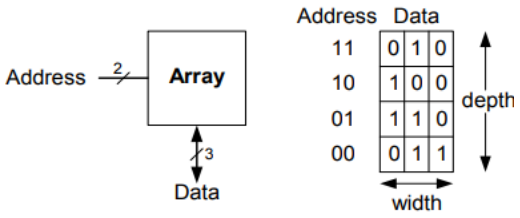
Goal: Efficiently store large amount of data
3 common types:

- DRAM
- SRAM
- ROM

M bit data stored at each N bit address

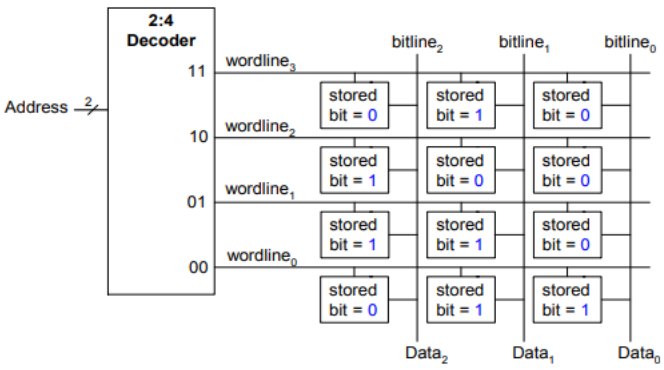


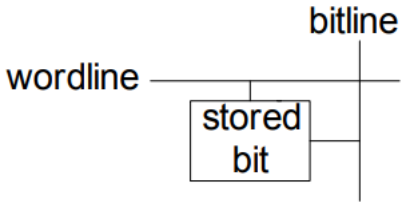
- 2 dimensional array of bit cells - each cell stores one bit
- N address bits and M data bits
  - 2^N rows and M columns
  - **Depth** - number of rows
  - **Width** - number of columns
  - **Array Size:** depth × width



1.1 Wordline

- A single row in memory array to be read/written
- Only one wordline high at once

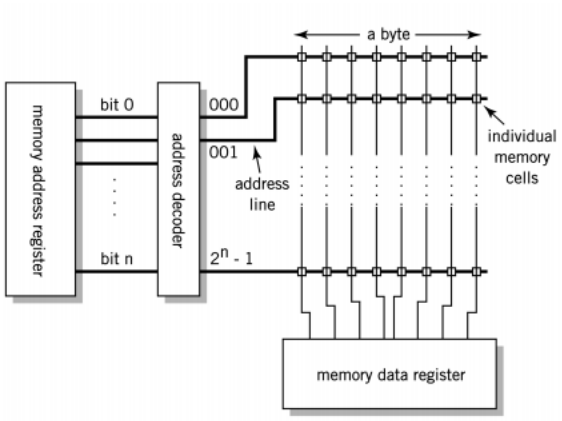


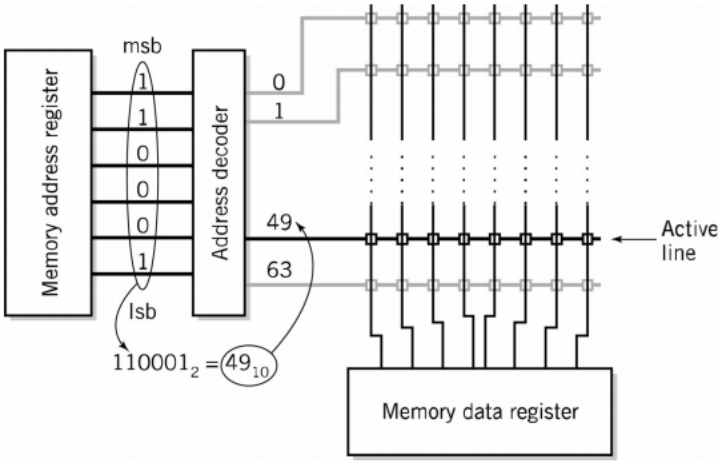


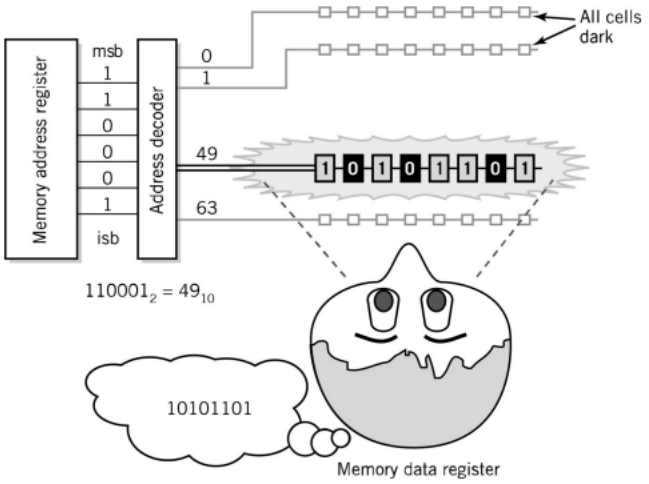
On memory read:

- If the wordline is active, the stored bit should drive the bitline value
- If the wordline is not active, the stored bit should not affect the bitline (leave it floating)

1.2 Memory Diagrams







1.3 Registers

Memory consists of latches, each holding a single bit value, at an address
Interface between CPU and memory

- MAR
- MDR

To speed access it is normal to address 8 bytes at a time, however the CPU can still isolate individual bytes when it needs to

MAR holds the 'open' address, the CPU activates the group of bytes required - the address line

MDR holds the activated data

1.4 Lines

Lines that control a memory cell

- Wordline/ Address Line
  - On when the computer is addressing the data in that cell
- Read/Write Line
  - Determines whether the data will be transferred to the MDR (write) or from the MDR (read) the cell
- Read occurs when
  - Address line = 1 AND R/W line =1
  - Bitlines are left floating at the MDR, so take their values (weakly) from the active memory cells in the address line.
- Write occurs when
  - Address line = 1 AND R/W line = 0
  - Bitlines are strongly driven by the MDR data, and overpower the weakly held bits in the active memory cells, overwriting the contents.

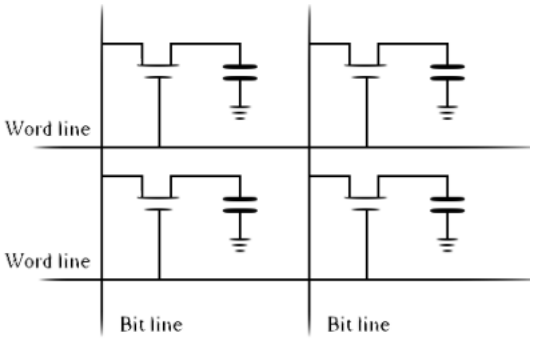
2 Types of memory

Characteristics of the memory depend on the design of the individual cells:

- RAM
  - Originally named thus to distinguish from serial memory, e.g. magnetic tape
  - Key feature: memory is volatile and is lost in power off.
  - DRAM - dynamic RAM, dense but needs refreshing
  - SRAM - static RAM less dense, but more stable.
- ROM
  - Originally was read only, and so named
  - Key feature: memory persists even during power off
  - Modern ROM memory, e.g. flash/SSD memory, can be written to

2.1 DRAM

- Each bit is one transistor and one capacitor.
- The capacitor is charged or discharged to represent a bit.
- Higher density, but slower.
- Capacitors discharge with time so need refreshing (reading and rewriting) every 64ms.

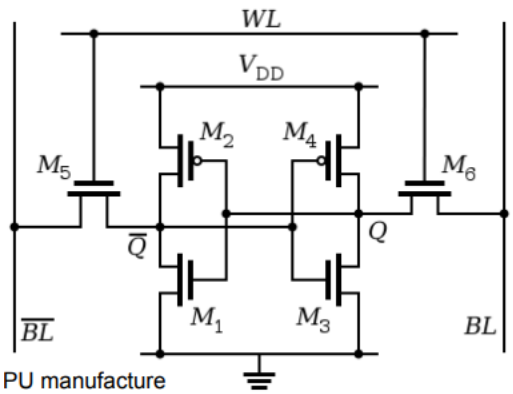


Dynamic Random Assess Memory (DRAM)

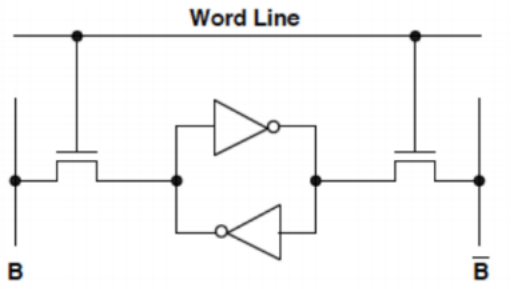
- Weakly driving the bitline as there is one capacitor driving the whole bitline

2.2 SRAM

- 6 (or more) transistor flip-flop based memory
- Low density but stable and fast.
- Doesn't require refreshing but loses memory when powered off.
- Can be easily incorporated into CPU manufacture



PU manufacture

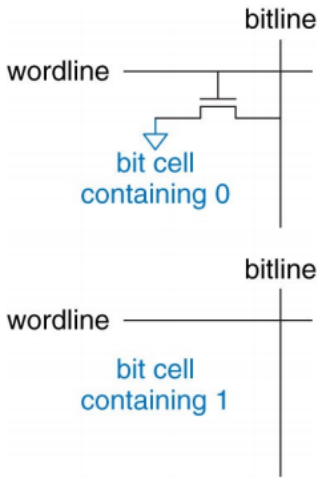


- Activating wl and connecting lhs to bit value, B will give the driven value
- To write, send a strong signal to the not gates and they will flip into the opposite state

2.3 ROM

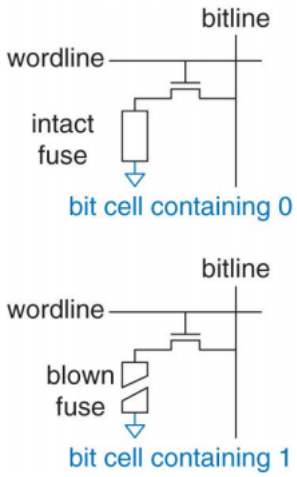
2.3.1 Basic ROM

- To read the bitline is weakly set to high The wordline is then activated
- If the cell contains a 0, the transistor connects the bitline to ground and the bitline goes low.
- If the bitline contains a 1, the bitline retains the weak high it was initialised with.



2.3.2 Programmable ROM

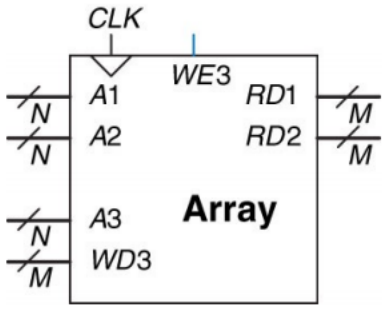
- All bit cells have a transistor. By apply sufficient voltage a fuse can be blown to disconnect the transistors from ground in cells meant to contain 1s.
- FLASH memory uses floating gate transistors instead, which can be electrically activated or deactivated. We won't go into details.



3 Multi Ported Memory

Memory that can access more than one address line at once.
Here the memory has inputs for

- 3 addresses
  - AD1 and AD2 are always read
  - Data appears at outputs RD1 and RD2
  - AD3 is address for optional writing
- Data line WD3 for data to be written
- WriteEnable line WE3
  - 0 – no write
  - 1 – write data WD3 to address A3
- Also clock input CLK
  - Will discuss timing elsewhere



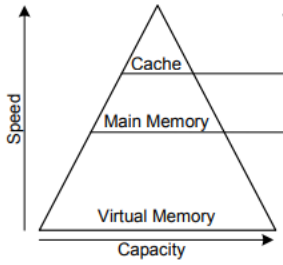
4 The MAR

The number of bits in the MAR determines how many different address locations can be decoded.

For an MAR of width k bits, the number of possible memory addresses is $M = 2^k$

Typical modern MAR is at least 32 bits wide, giving a 4 gigabytes memory.
Some may be 64 bits wide, allowing 16 exabytes of memory!

5 Memory Hierarchy



Technology	Price / GB	Access Time (ns)	Bandwidth (GB/s)
SRAM	$10,000	1	25+
DRAM	$10	10 - 50	10
SSD	$1	100,000	0.5
HDD	$0.1	10,000,000	0.1

- Ideal memory (Fast, Cheap and Large capacity), is impossible.
- We use a hierarchy of memories to give the effect of ideal memory.

6 Memory Speed

Memory is slow compared to CPU processing speeds!
- 2Ghz CPU = 1 cycle in ½ of a billionth of a second
- 70ns DRAM = 1 access in 70 millionth of a second
- Registers are much faster – they are in the CPU itself; memory is on a separate chip

Methods to improve memory access
- **Wide Path Memory Access**
  Retrieve multiple bytes instead of 1 byte at a time
- **Memory Interleaving**
  Partition memory into subsections, each with its own address register and data register
- **Cache Memory**

7 Wide Path Memory Access

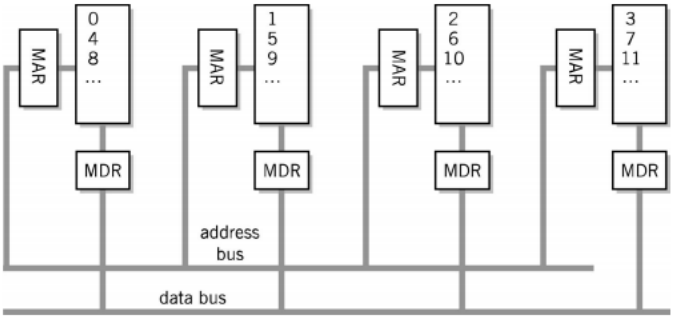
When calling or storing memory, don't call up just 1 bit or byte. Most modern CPUs load 8 bytes (64 bits) at a time.

- Requires a **wider memory-data bus** and **larger MDR**
- The 8 bytes can be separated and used appropriately in the CPU - **using more circuitry**
- **Diminishing returns** – widening the path further would involve more complexity in the CPU and extra bytes are less likely to be used often.

A common theme we see in digital electronics:
the trade off between **higher speed for specific operations** and **increased hardware complexity** in the CPU.

8 Memory Interleaving

- **Divide the memory** into separate chunks that can be accessed simultaneously.
- **Interleave**, so consecutive addresses can be accessed simultaneously.
- Useful for **parallelisation** / **pipelining** (more later).



9 Cache Memory

- Even the fastest hard disk has an access time of about 10 milliseconds.
- 2Ghz CPU waiting 10 milliseconds wastes **20 million clock cycles!**

Idea: Preload some special fast memory with the data you are likely to use soon.

What data are you likely to use soon?

Typically, data near addresses you have recently accessed.

What is 'fast memory'

Memory on the CPU die itself – not a separate RAM chip.

More expensive SRAM memory

10 Locality

What data is 'near'?

Temporal Locality:

Locality in time

If data used recently, likely to use it again soon

How to exploit: keep recently accessed data in higher levels of memory hierarchy

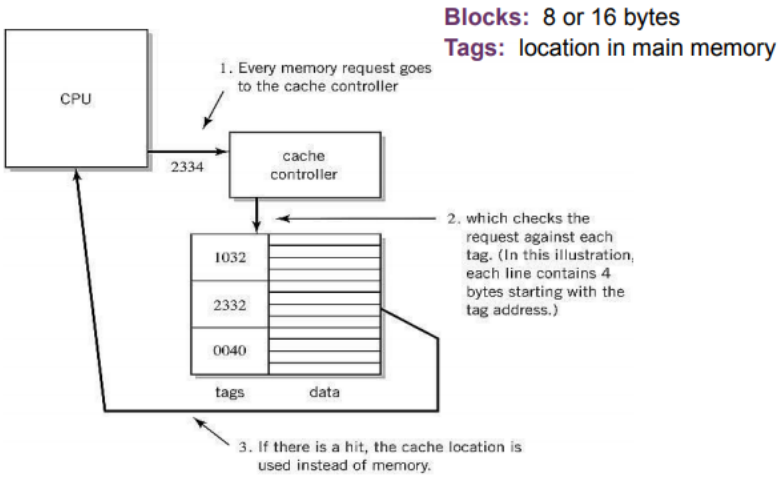
Spatial Locality:

Locality in space

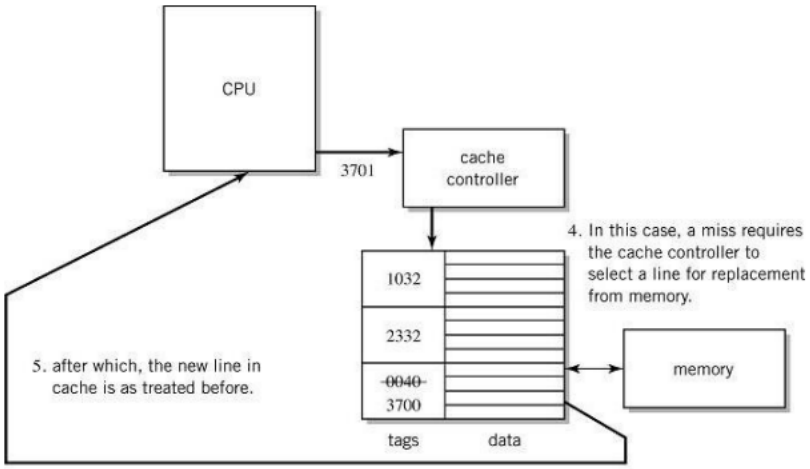
If data used recently, likely to use nearby data soon

How to exploit: when access data, bring nearby data into higher levels of memory hierarchy too

11 Cache Hit



12 Cache Miss



13 Cache

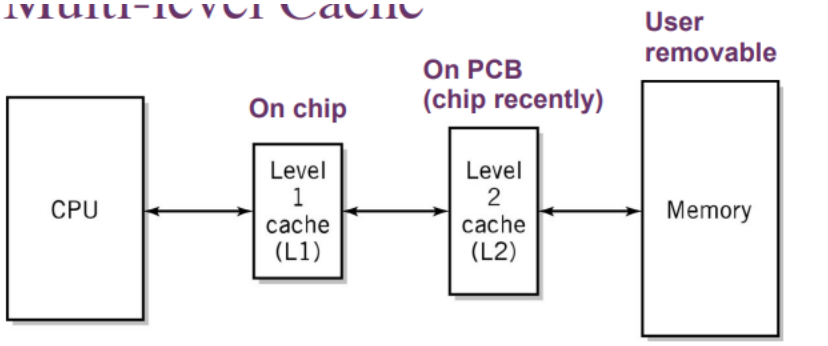
Hit Ratio: ratio of hits to total requests

- Hit ratios of 90% common
- 50%+ improved execution speed
- Locality of reference is why caching works:
  - Most memory references confined to small region of memory at any given time. E.g.
    - A well-written program in small loop, procedure or function
    - Data in an array
    - Variables stored together

Synchronizing cache and memory:

- Write through – writes immediately go through to memory
- Write back – writes only sent through when block flushed

14 Multi Level Cache



Level 2 cache – larger, further away and slower, but still better than main memory.

Intel Core DUO: 64KB level 1 cache, 2MB level 2 cache, up to 8GB memory.

Multi-core CPUs: each core has its own L1 and L2 cache, and there is typically a shared L3 cache.