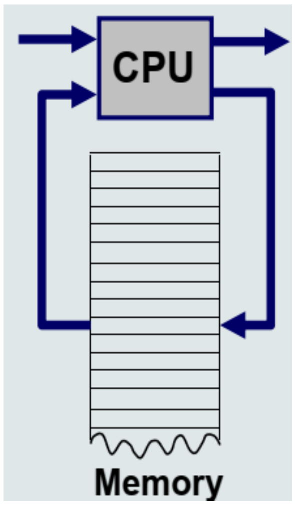
Chapter 3

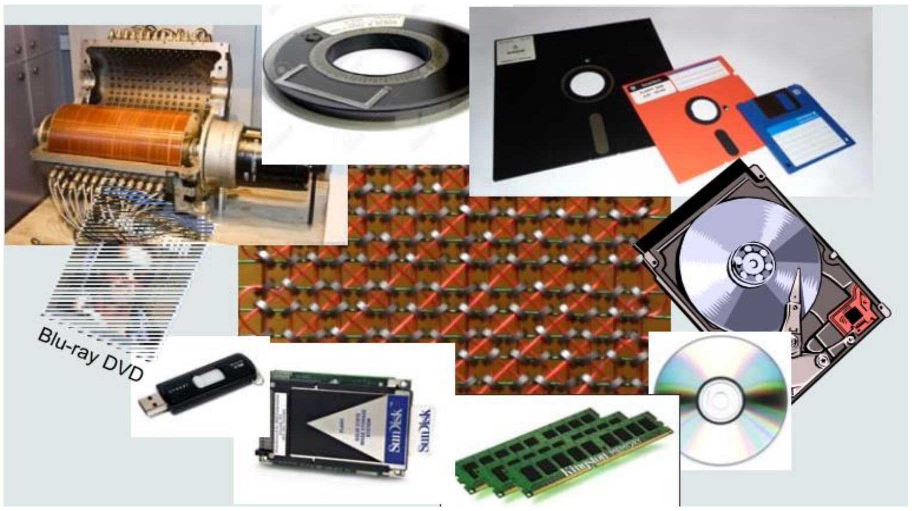
COMPUTER MEMORY
Part 3
Internal memory

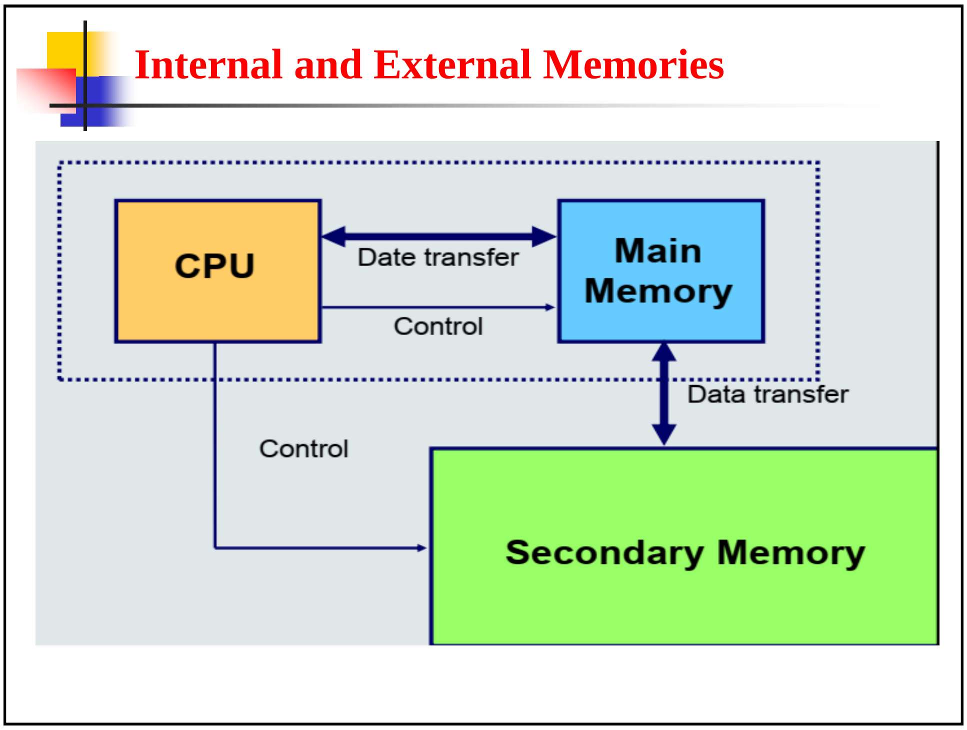
Basic Principles of Computers

- Virtually all modern computer designs are based on the von Neumann architecture principles:
- Data and instructions are stored in a single read/write memory.
- The contents of this memory are addressable by location, without regard to what are stored there.
- Instructions are executed sequentially (from one instruction to the next) unless the order is explicitly modified

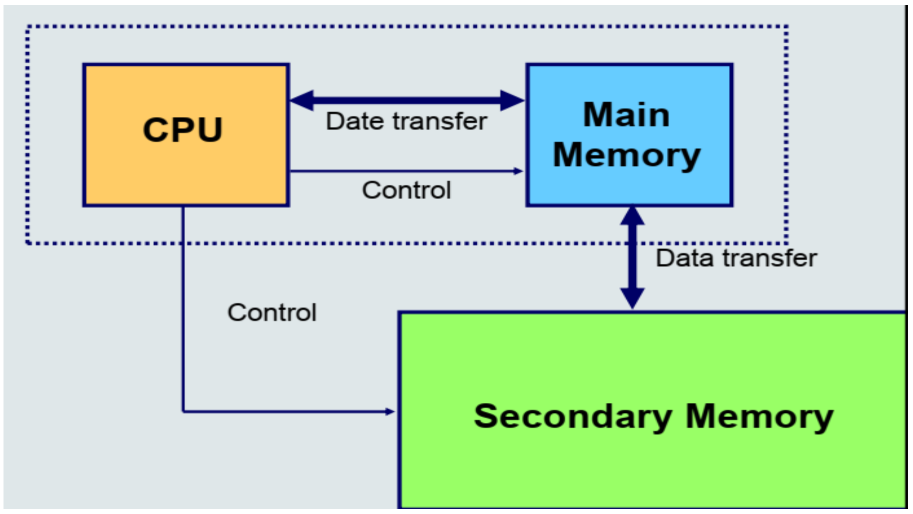


Many Different Technologies

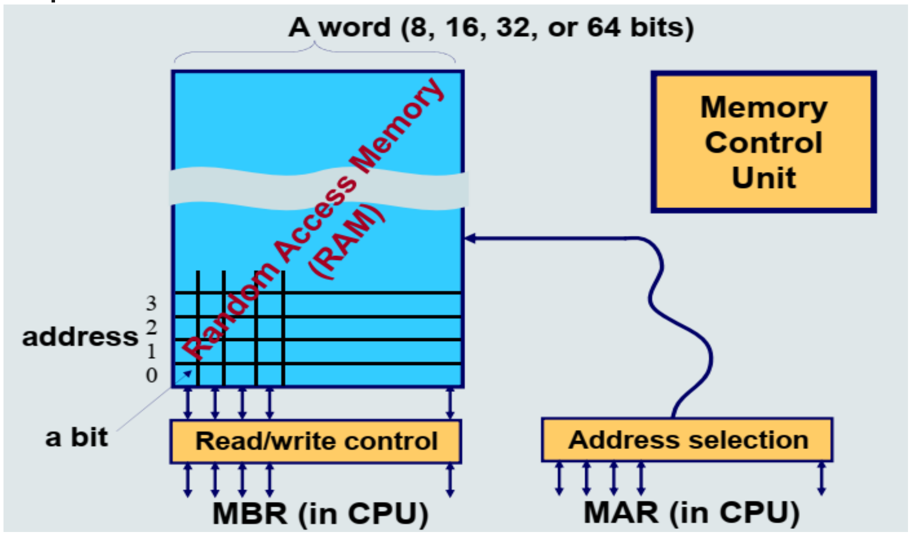




Internal and External Memories

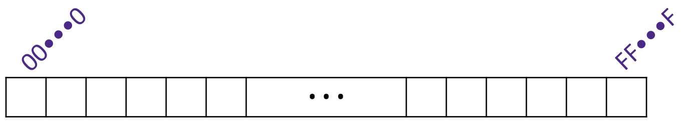


Main Memory Model



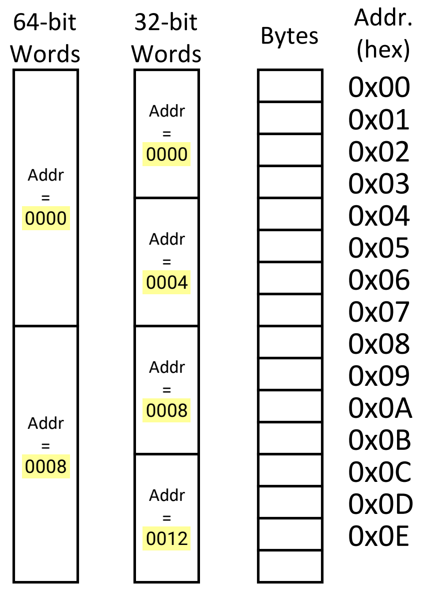
Byte-Oriented Memory Organization

- Conceptually, memory is a single, large array of bytes, each with a unique *address* (index)
  - The value of each byte in memory can be read and written
- Programs refer to bytes in memory by their *addresses*
  - Domain of possible addresses = *address space*
- But not all values fit in a single byte… (*e.g.* 410)
  - Many operations actually use multi-byte values
- We can store addresses as data to "remember" where other data is in memory



Word-Oriented Memory Organization

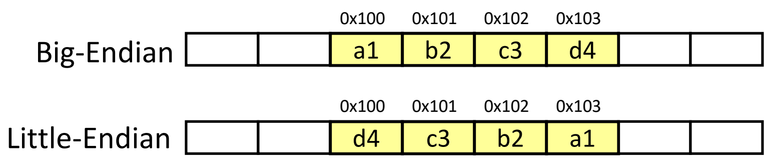
- Addresses still specify locations of *bytes* in memory
  - Addresses of successive words differ by word size (in bytes): *e.g.* 4 (32-bit) or 8 (64-bit)
  - Address of word 0, 1, … 10?
- Address of word
  = address of *first* byte in word
  - The address of *any* chunk of memory is given by the address of the first byte
  - ***Alignment***



Byte Ordering

- How should bytes within a word be ordered *in memory?*
  - **Example:** store the 4-byte (32-bit) `int`:
    `0x a1 b2 c3 d4`
- By convention, ordering of bytes called *endianness*
  - The two options are big-endian and little-endian
  - Based on *Gulliver's Travels*: tribes cut eggs on different sides (big, little)

- Big-endian (SPARC, z/Architecture)
  - Least significant byte has highest address
- Little-endian (x86, x86-64)
  - Least significant byte has lowest address
- Bi-endian (ARM, PowerPC)
  - Endianness can be specified as big or little
- **Example:** 4-byte data 0xa1b2c3d4 at address 0x100

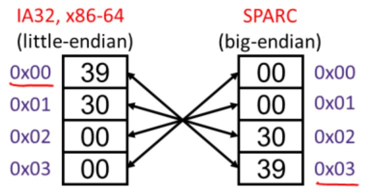


Byte Ordering Examples

```
Decimal:           12345
Binary:   0011 0000 0011 1001
Hex:        3    0    3    9
```

4 bytes

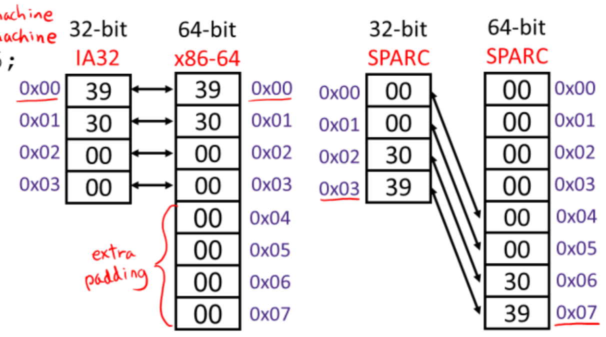
```
int x = 12345;
// or x = 0x3039;
```



4 bytes on 32-bit machine
8 bytes on 64-bit machine

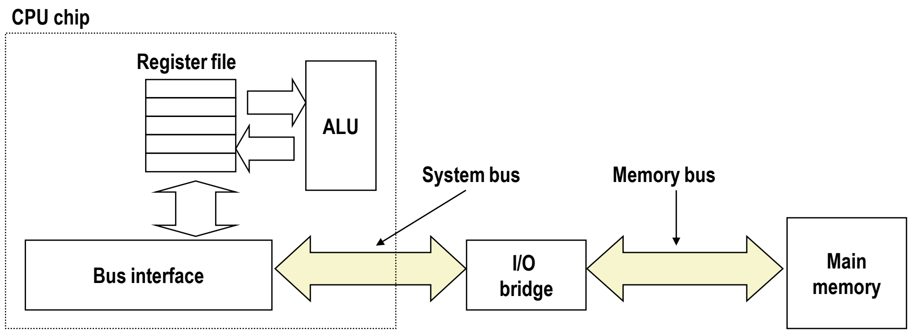
```
long int y = 12345;
// or y = 0x3039;
```

(A `long int` is the size of a word)



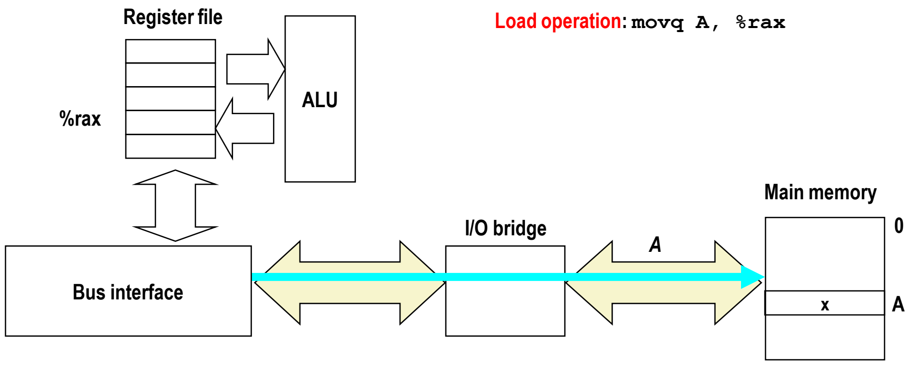
Traditional Bus Structure Connecting CPU and Memory

- A bus is a collection of parallel wires that carry address, data, and control signals.
- Buses are typically shared by multiple devices.



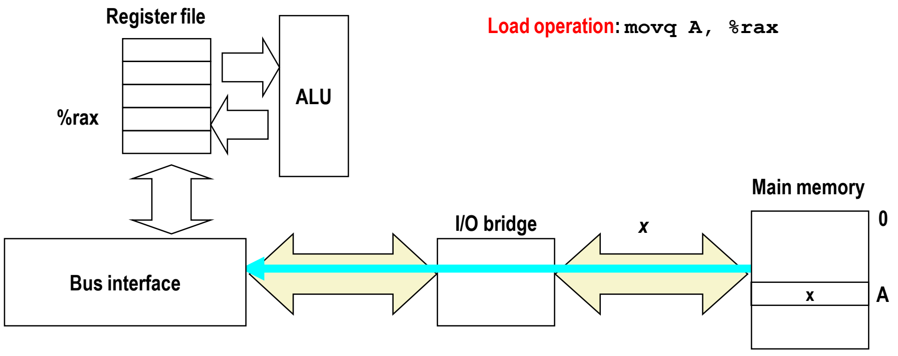
Memory Read Transaction (1)

- CPU places address A on the memory bus.



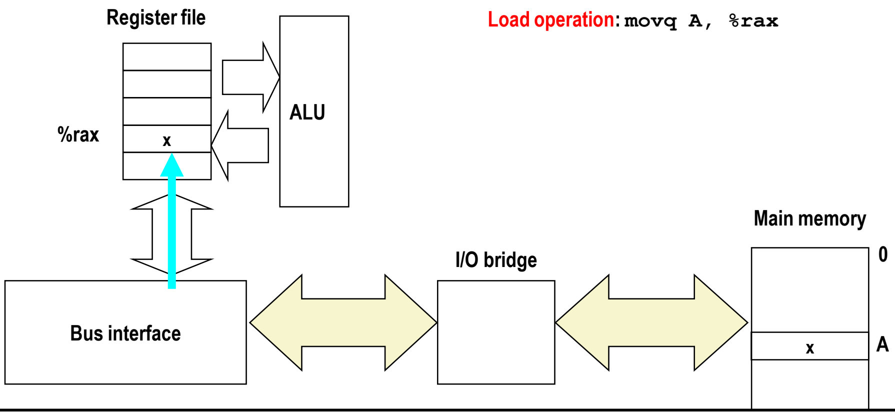
Memory Read Transaction (2)

- Main memory reads A from the memory bus, retrieves word x, and places it on the bus.



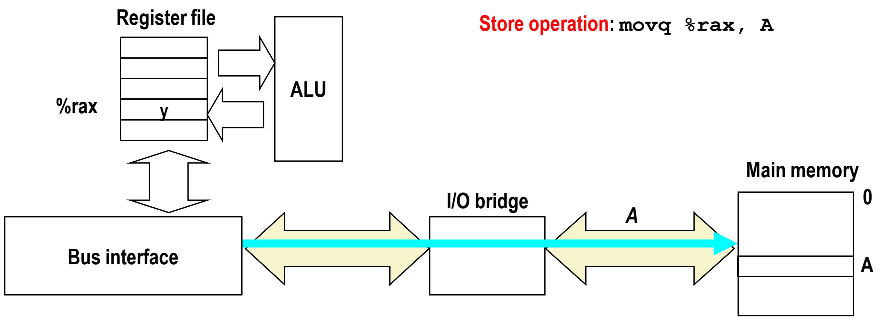
Memory Read Transaction (3)

- CPU read word x from the bus and copies it into register %rax.



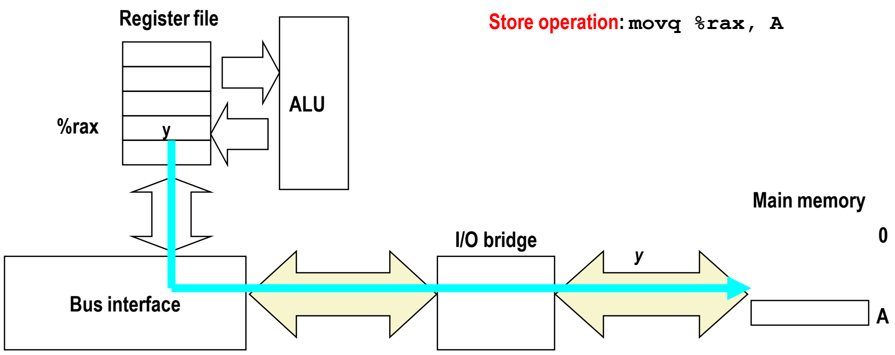
Memory Write Transaction (1)

- CPU places address A on bus. Main memory reads it and waits for the corresponding data word to arrive.



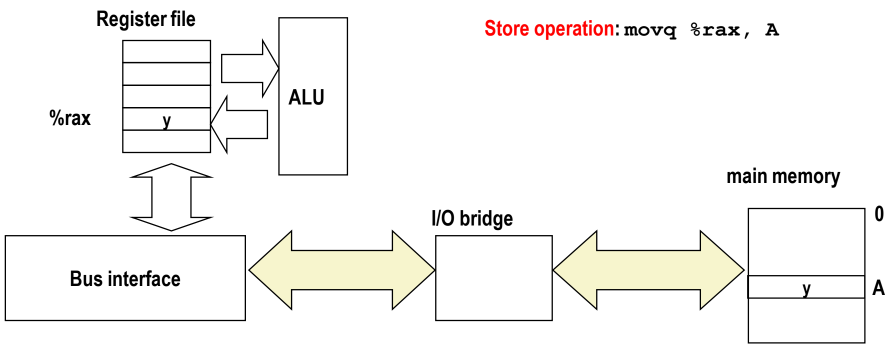
Memory Write Transaction (2)

- CPU places data word y on the bus.



Memory Write Transaction (3)

- Main memory reads data word y from the bus and stores it at address A.



Physical types

- Semiconductor
  - RAM
- Magnetic
  - Disk & Tape
- Optical
  - CD & DVD
- Others
  - Bubble
  - Hologram

Storing data in main memory

- Possible types of memories:
- ROM: read-only
  - Classical ROM: the content is stored during the manufacturing process
  - PROM: one-time programmable
  - EPROM: can be erased using ultraviolet light
  - Etc.
- SRAM: Static Random Access Memory
  - Can be read and modified any time
  - It preserves data while power supply is present
- DRAM: Dynamic Random Access Memory
  - Can be read and modified any time
  - It forgets its content! Needs to be refreshed periodically

Nonvolatile Memories

- DRAM and SRAM are volatile memories
  - Lose information if powered off.
- Nonvolatile memories retain value even if powered off
  - Read-only memory (ROM): programmed during production
  - Programmable ROM (PROM): can be programmed once
  - Eraseable PROM (EPROM): can be bulk erased (UV, X-Ray)
  - Electrically eraseable PROM (EEPROM): electronic erase capability
  - Flash memory: EEPROMs. with partial (block-level) erase capability
    - Wears out after about 100,000 erasings
- Uses for Nonvolatile Memories
  - Firmware programs stored in a ROM (BIOS, controllers for disks, network cards, graphics accelerators, security subsystems,…)
  - Solid state disks (replace rotating disks in thumb drives, smart phones, mp3 players, tablets, laptops,…)
  - Disk caches

Random-Access Memory (RAM)

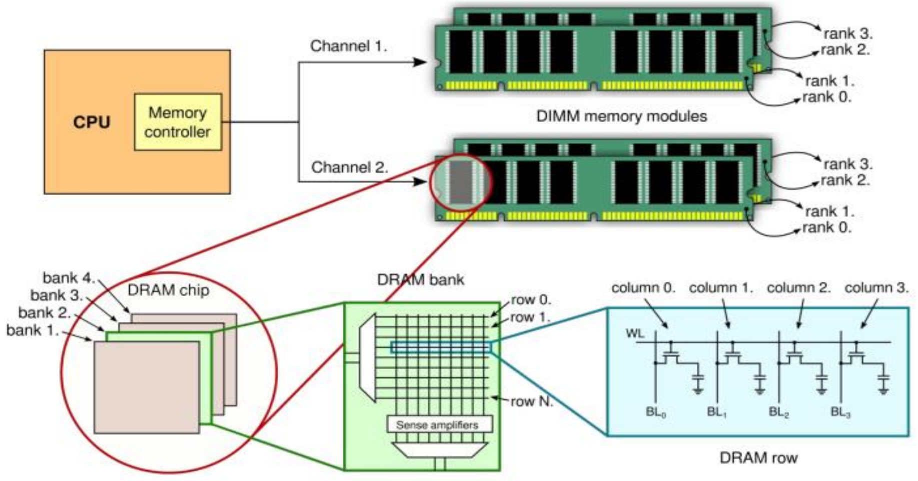
- Key features
  - RAM is traditionally packaged as a chip
  - Basic storage unit is normally a cell (one bit per cell)
  - Multiple RAM chips form a memory
- Static RAM (SRAM)
  - Each cell stores a bit with a four- or six-transistor circuit
  - Retains value indefinitely, as long as it is kept powered
  - Relatively insensitive to electrical noise (EMI), radiation, etc.
  - Faster and more expensive than DRAM
- Dynamic RAM (DRAM)
  - Each cell stores bit with a capacitor; transistor is used for access
  - Value must be refreshed every 10-100 ms
  - More sensitive to disturbances (EMI, radiation,…) than SRAM
  - Lower and cheaper than SRAM

SRAM vs DRAM Summary

- EDC = error detection and correction
  - To cope with noise, etc.

	Trans. per bit	Access time	Needs refresh?	Needs EDC?	Cost	Applications
SRAM	4 or 6	1X	No	Maybe	100x	Cache memories
DRAM	1	10X	Yes	Yes	1X	Main memories, frame buffers

- How to create a memory system for the computer
  - It must be: cheap, large, low latency, high throughput



DRAM

- Bits stored as charge in capacitors
- Charges leak
- Need refreshing even when powered
- Simpler construction
- Smaller per bit
- Less expensive
- Need refresh circuits
- Slower
- Main memory
- Essentially analogue
  - Level of charge determines value

Static RAM

- Bits stored as on/off switches
- No charges to leak
- No refreshing needed when powered
- More complex construction
- Larger per bit
- More expensive
- Does not need refresh circuits
- Faster
- Cache
- Digital
  - Uses flip-flops

SRAM v DRAM

- Both volatile
  - Power needed to preserve data
- Dynamic cell
  - Simpler to build, smaller
  - More dense
  - Less expensive
  - Needs refresh
  - Larger memory units
- Static
  - Faster
  - Cache

Summary: DRAM vs. SRAM

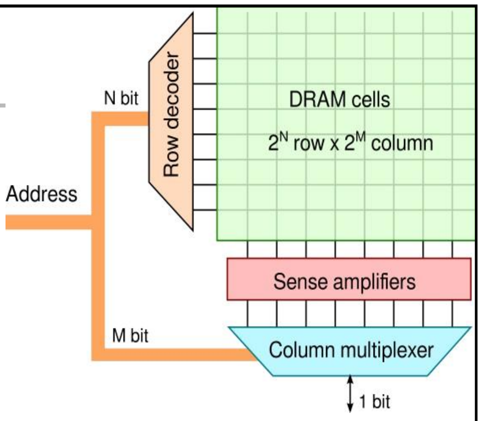
- DRAM (Dynamic RAM)
- Used mostly in main mem.
- Capacitor + 1 transistor/bit
- Need refresh every 4-8 ms
  - 5% of total time
- Read is destructive (need for write-back)
- Access time < cycle time (because of writing back)
- Density (25-50):1 to SRAM
- Address lines multiplexed
  - pins are scarce!

- SRAM (Static RAM)
- Used mostly in caches (I, D, TLB, BTB)
- 1 flip-flop (4-6 transistors) per bit
- Read is not destructive
- Access time = cycle time
- Speed (8-16):1 to DRAM
- Address lines not multiplexed
  - high speed of decoding imp.

Chip Organization

- Chip capacity (= number of data bits)
  - tends to quadruple
  - 1K, 4K, 16K, 64K, 256K, 1M, 4M, …
- In early designs, each data bit belonged to a different address (x1 organization)
- Starting with 1Mbit chips, wider chips (4, 8, 16, 32 bits wide) began to appear
  - Advantage: Higher bandwidth
  - Disadvantage: More pins, hence more expensive packaging

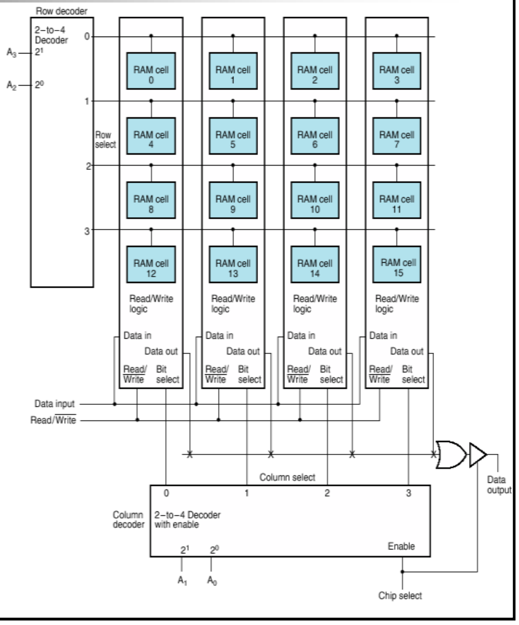
DRAM bank



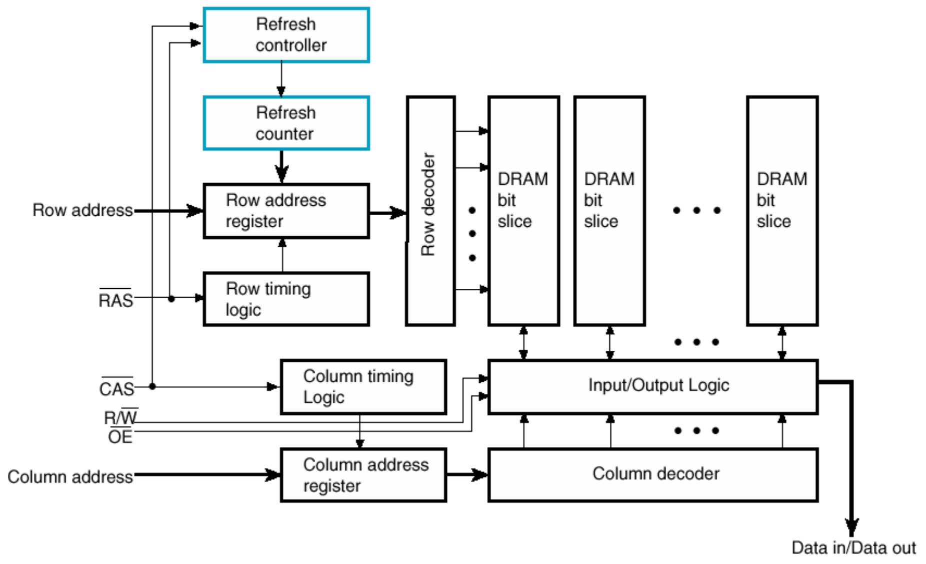
- Structure:
  DRAM cells in a 2D grid
- Each cell in a row shares the same word line
- Each cell in a column shares the same bit line
- Reading:
  - The **row decoder** selects (activates) a row
  - The **sense amplifiers** detect and store the bits of the row
  - The **column multiplexer** selects the desired column from the row
- Two-phase operations:
  - To reduce the width of the address bus
  - Address bus: row address → wait → address bus: column address→ data bus: the desired data

16 X 1 as 4 X 4 Array

- Two decoders
  - Row
  - Column
- Address just broken up
- Not visible from outside

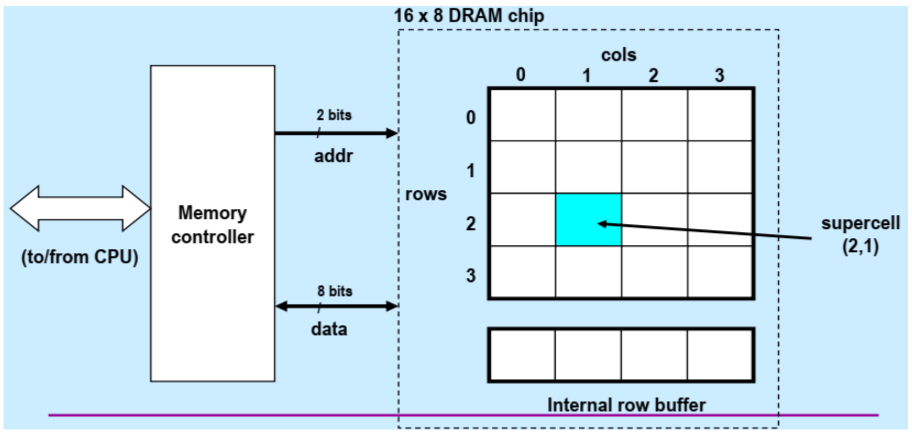


DRAM Logical Diagram



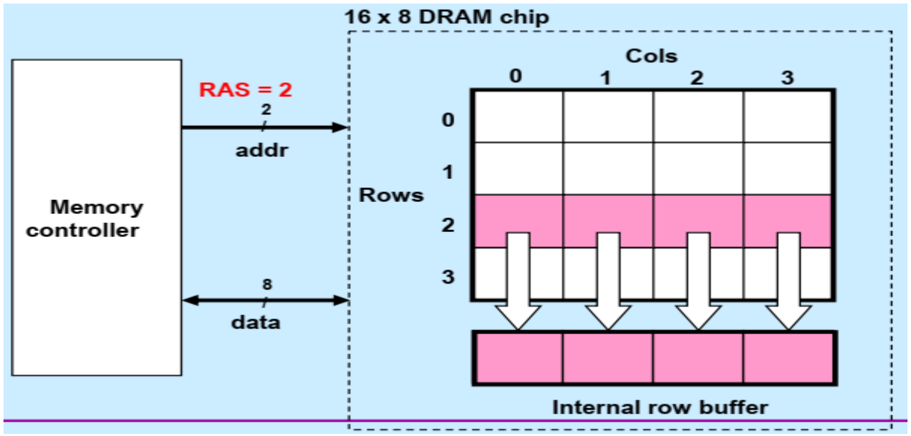
Conventional DRAM Organization

- d x w DRAM:
  - dw total bits organized as d supercells of size w bits



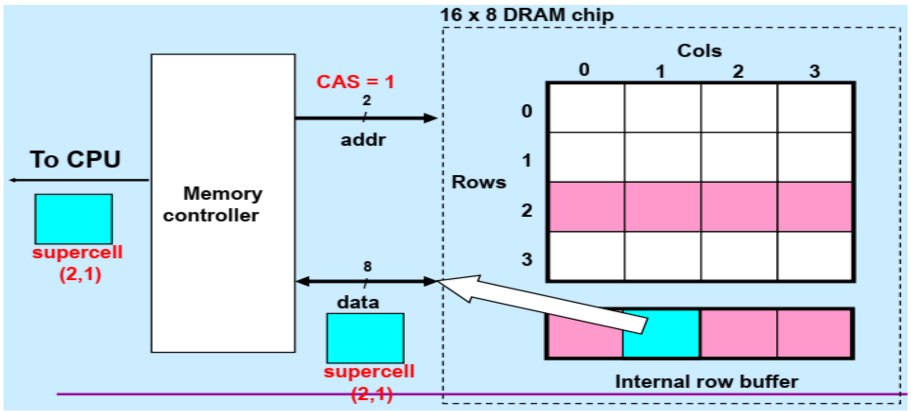
Reading DRAM Supercell (2,1)

- Step 1(a): row access strobe (RAS) selects row 2
- Step 1(b): row 2 copied from DRAM array to row buffer



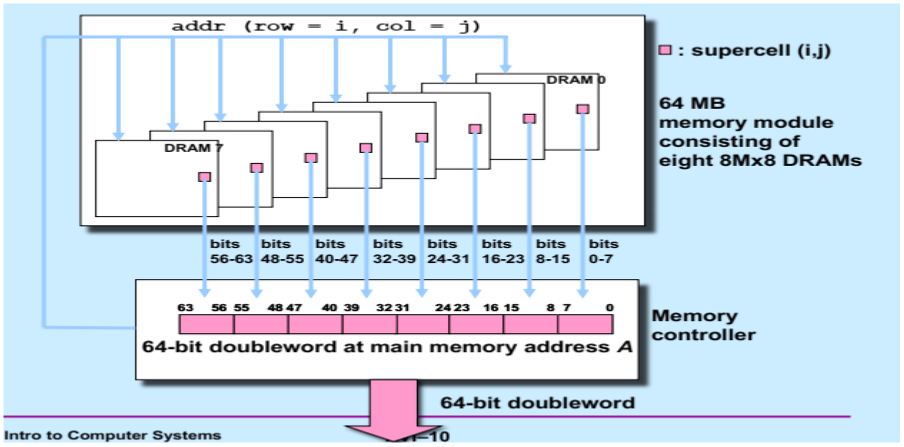
Reading DRAM Supercell (2,1)

- Step 2(a): column access strobe (CAS) selects column 1
- Step 2(b): supercell (2,1) copied from buffer to data lines, and eventually back to the CPU



Memory Modules

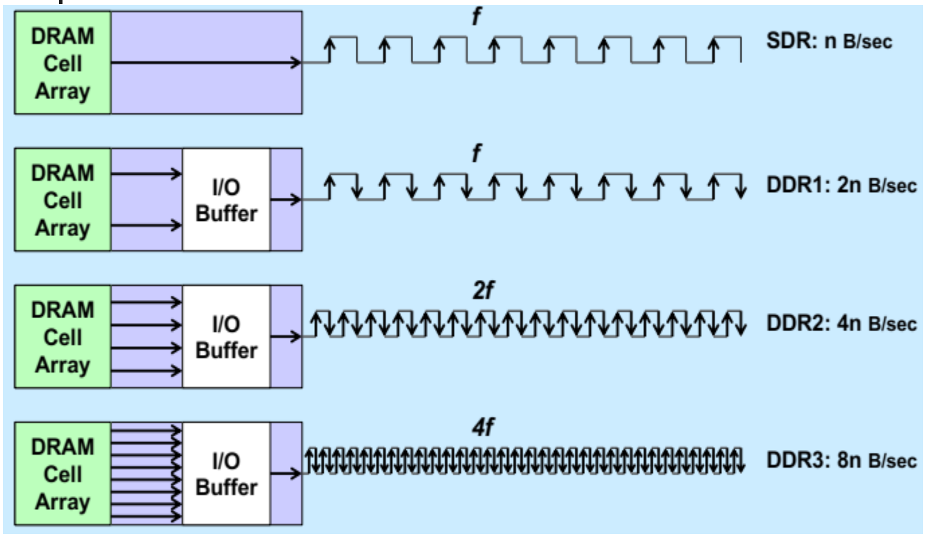
- Combine some of chip to create memory module



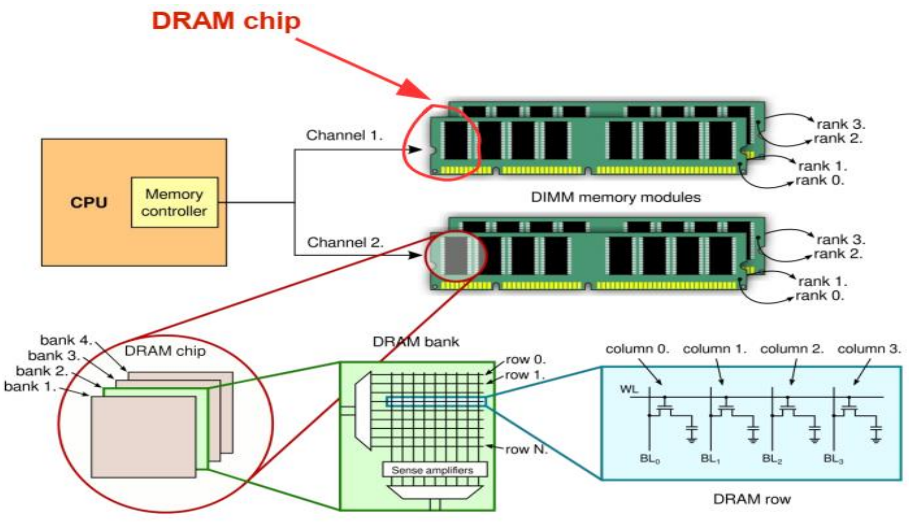
Enhanced DRAMs

- Basic DRAM cell has not changed since its invention in 1966
  - Commercialized by Intel in 1970
- DRAMs with better interface logic and faster I/O:
  - Synchronous DRAM (SDRAM)
    - Uses a conventional clock signal instead of asynchronous control
    - Allows reuse of the row addresses (e.g., RAS, CAS, CAS, CAS)
  - Double data-rate synchronous DRAM (DDR SDRAM)
    - DDR1 : twice as fast
    - DDR2 : four times as fast
    - DDR3 : eight times as fast

Enhanced DRAMs



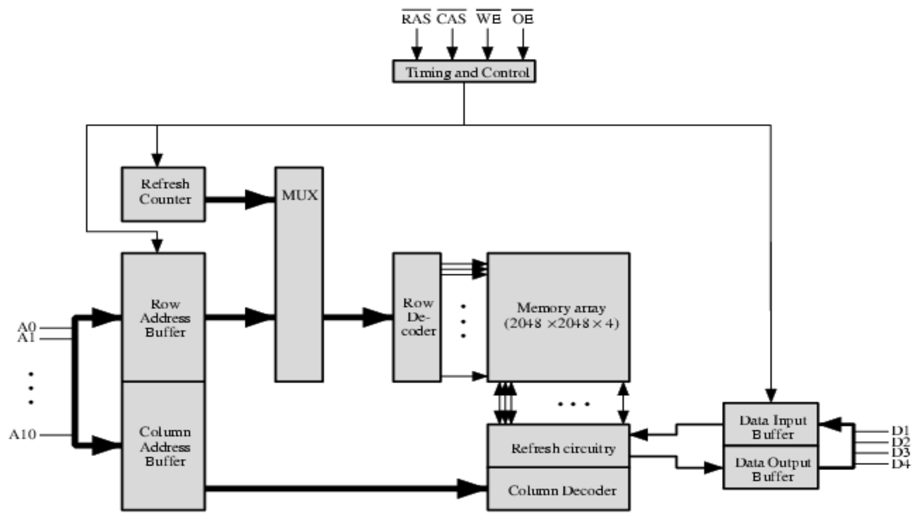
DRAM chips



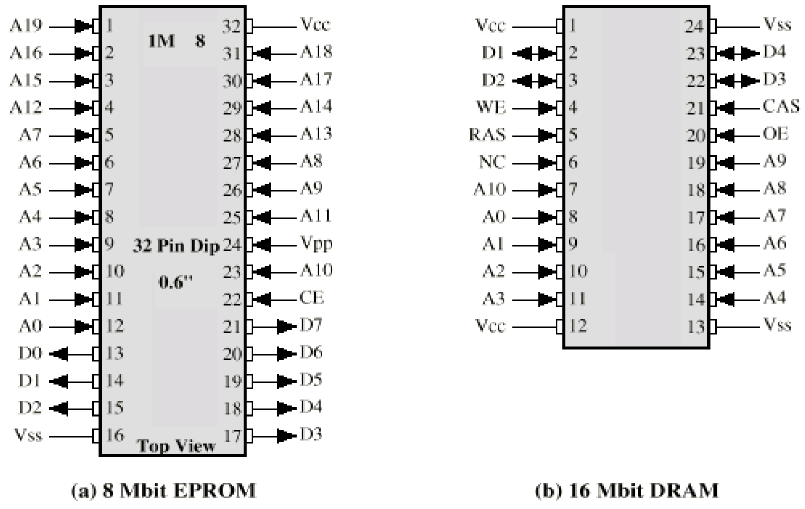
Organisation in detail

- A 16Mbit chip can be organised as 1M of 16 bit words
- A bit per chip system has 16 lots of 1Mbit chip with bit 1 of each word in chip 1 and so on
- A 16Mbit chip can be organised as a 2048 x 2048 x 4bit array
  - Reduces number of address pins
    - Multiplex row address and column address
    - 11 pins to address (2^{11}=2048)
    - Adding one more pin doubles range of values so x4 capacity

Typical 16 Mb DRAM (4M x 4)



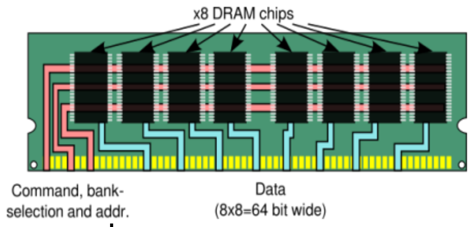
Packaging



1M 8
32 Pin Dip
0.6"
Top View
A19 1, A16 2, A15 3, A12 4, A7 5, A6 6, A5 7, A4 8, A3 9, A2 10, A1 11, A0 12, D0 13, D1 14, D2 15, Vss 16
Vcc 32, A18 31, A17 30, A14 29, A13 28, A8 27, A9 26, A11 25, Vpp 24, A10 23, CE 22, D7 21, D6 20, D5 19, D4 18, D3 17
(a) 8 Mbit EPROM
Vcc 1, D1 2, D2 3, WE 4, RAS 5, NC 6, A10 7, A0 8, A1 9, A2 10, A3 11, Vcc 12
Vss 24, D4 23, D3 22, CAS 21, OE 20, A9 19, A8 18, A7 17, A6 16, A5 15, A4 14, Vss 13
(b) 16 Mbit DRAM

DRAM MEMORY MODULE

- A memory module consists of DRAM chips
- Command lines, bank selection lines, address lines: shared
- Data lines: concatenated

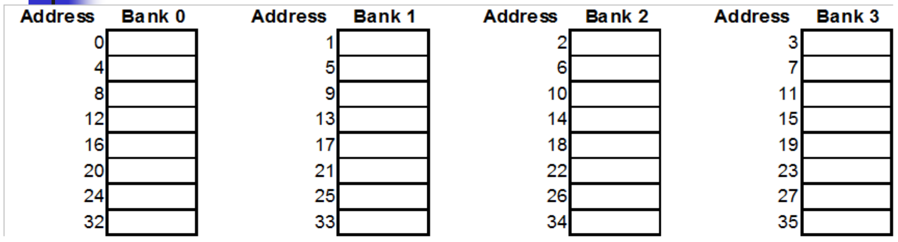


- Each chip receives all commands
- Effect:
  - Throughput increases 8x
  - Delay: the same

Memory Interleaving

- Goal: Try to take advantage of bandwidth of multiple DRAMs in memory system
- Memory address *A* is converted into (*b*,*w*) pair, where
  - *b* = bank index
  - *w* = word index within bank
- Logically a wide memory
  - Accesses to *B* banks staged over time to share internal resources such as memory bus
- Interleaving can be on
  - Low-order bits of address (cyclic)
    - *b* = *A* mod *B*, *w* = *A* div *B*
  - High-order bits of address (block)
  - Combination of the two (block-cyclic)

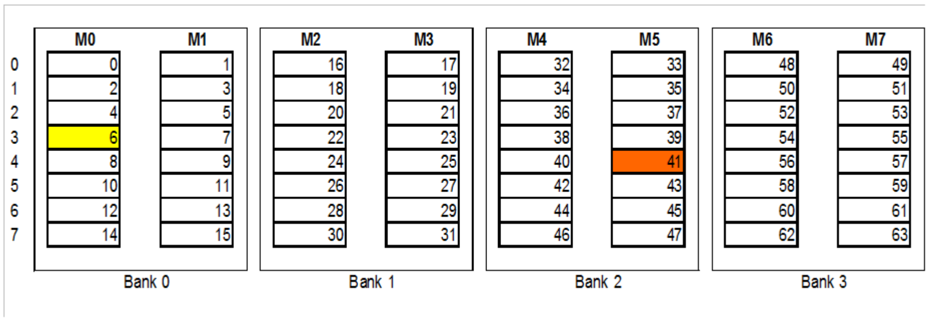
Low-order Bit Interleaving



	1	2	3	4	5	6	7	8	9	10	11	12
Bank 0	0	0	0		4	4	4					
Bank 1		1	1	1		5	5	5				
Bank 2			2	2	2		6	6	6			
Bank 3				3	3	3		7	7	7		
Bus				0	1	2	3	4	5	6	7	

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Bank 0	0	0	0	0	0	4	4	4	4	4				
Bank 1		1	1	1	1	1	5	5	5	5	5			
Bank 2			2	2	2	2	2	6	6	6	6	6		
Bank 3				3	3	3	3	3	7	7	7	7	7	
Bus						0	1	2	3		4	5	6	7

Mixed Interleaving



- Memory address register is 6 bits wide
  - Most significant 2 bits give bank address
  - Next 3 bits give word address within bank
  - LSB gives (parity of) module within bank
- $6 = 000110_2 = (00, 011, 0) = (0, 3, 0)$
- $41 = 101001_2 = (10, 100, 1) = (2, 4, 1)$