

Chapter 6  
The Memory Hierarchy

Agenda

- Storage technologies and trends
- Locality of reference
- Caching in the memory hierarchy

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.
- RAM comes in two varieties:
  - SRAM (Static RAM)
  - DRAM (Dynamic RAM)

SRAM vs DRAM Summary

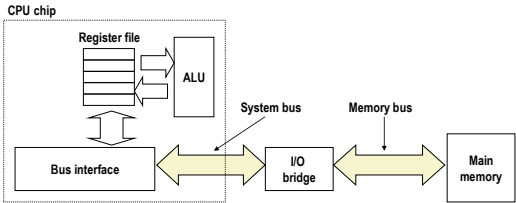
	Trans. per bit	Access time	Needs refresh?	Needs EDC?	Cost	Applications
SRAM	4 or 6	1X	No	Maybe	100x	Cache memories
DRAM	1	10X	Yes	Yes	1X	Main memories, frame buffers

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
  - Erasable PROM (EPROM): can be bulk erased (UV, X-Ray)
  - Electrically erasable 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

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.

Register file

%rax

ALU

Bus interface

I/O bridge

Main memory

Load operation: `movq A, %rax`

0

A

x

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

7

Memory Read Transaction (2)

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

Register file

%rax

ALU

Bus interface

I/O bridge

Main memory

Load operation: `movq A, %rax`

0

A

x

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

8

Memory Read Transaction (3)

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

Register file

%rax

ALU

Bus interface

I/O bridge

Main memory

Load operation: `movq A, %rax`

0

A

x

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

9

Memory Write Transaction (1)

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

Register file

%rax

ALU

Bus interface

I/O bridge

Main memory

Store operation: `movq %rax, A`

0

A

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

10

Memory Write Transaction (2)

■ CPU places data word y on the bus.

Register file

%rax

ALU

Bus interface

I/O bridge

Main memory

Store operation: `movq %rax, A`

0

A

y

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

11

Memory Write Transaction (3)

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

Register file

%rax

ALU

Bus interface

I/O bridge

main memory

Store operation: `movq %rax, A`

0

A

y

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

12

©R.A. Rutenbar, 2001

2

### What's Inside A Disk Drive?

Labels: Arm, Spindle, Platters, Actuator, SCSI connector, Electronics (including a processor and memory!).

Image courtesy of Seagate Technology

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

### Disk Geometry

- Disks consist of **platters**, each with two **surfaces**.
- Each surface consists of concentric rings called **tracks**.
- Each track consists of **sectors** separated by **gaps**.

Labels: Tracks, Surface, Spindle, Track k, Gaps, Sectors.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

### Disk Geometry (Multiple-Platter View)

- Aligned tracks form a **cylinder**.

Labels: Surface 0, Surface 1, Surface 2, Surface 3, Surface 4, Surface 5, Cylinder k, Platter 0, Platter 1, Platter 2, Spindle.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

### Disk Capacity

- **Capacity**: maximum number of bits that can be stored.
  - Vendors express capacity in units of gigabytes (GB), where 1 GB = 10<sup>9</sup> Bytes.
- Capacity is determined by these technology factors:
  - **Recording density** (bits/in): number of bits that can be squeezed into a 1 inch segment of a track.
  - **Track density** (tracks/in): number of tracks that can be squeezed into a 1 inch radial segment.
  - **Areal density** (bits/in<sup>2</sup>): product of recording and track density.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

### Recording zones

- Modern disks partition tracks into disjoint subsets called **recording zones**.
  - Each track in a zone has the same number of sectors, determined by the circumference of innermost track.
  - Each zone has a different number of sectors/track, outer zones have more sectors/track than inner zones.
  - So we use **average** number of sectors/track when computing capacity.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

### Computing Disk Capacity

Capacity = (# bytes/sector) x (avg. # sectors/track) x (# tracks/surface) x (# surfaces/platter) x (# platters/disk)

Example:

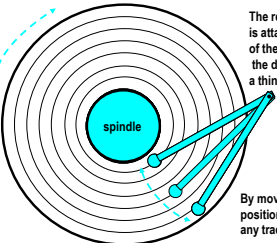
- 512 bytes/sector
- 300 sectors/track (on average)
- 20,000 tracks/surface
- 2 surfaces/platter
- 5 platters/disk

Capacity = 512 x 300 x 20000 x 2 x 5  
= 30,720,000,000  
= 30.72 GB

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

Disk Operation (Single-Platter View)

The disk surface spins at a fixed rotational rate



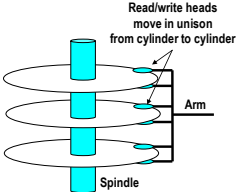
The read/write head is attached to the end of the arm and flies over the disk surface on a thin cushion of air.

By moving radially, the arm can position the read/write head over any track.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

19

Disk Operation (Multi-Platter View)



Read/write heads move in unison from cylinder to cylinder

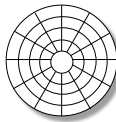
Arm

Spindle

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

20

Disk Structure - top view of single platter



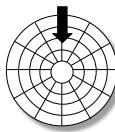
Surface organized into tracks

Tracks divided into sectors

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

21

Disk Access

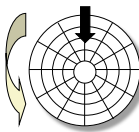


Head in position above a track

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

22

Disk Access

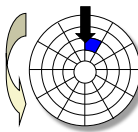


Rotation is counter-clockwise

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

23

Disk Access – Read

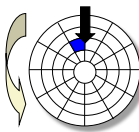


About to read blue sector

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

24

Disk Access – Read



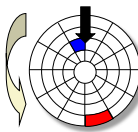
After BLUE read

After reading blue sector

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

25

Disk Access – Read



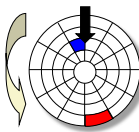
After BLUE read

Red request scheduled next

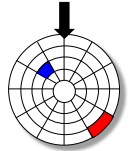
Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

26

Disk Access – Seek



After BLUE read



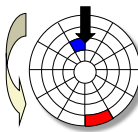
Seek for RED

Seek to red's track

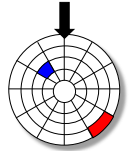
Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

27

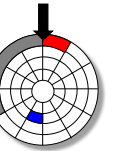
Disk Access – Rotational Latency



After BLUE read



Seek for RED



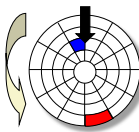
Rotational latency

Wait for red sector to rotate around

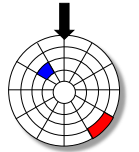
Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

28

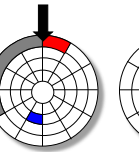
Disk Access – Read



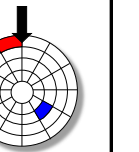
After BLUE read



Seek for RED



Rotational latency



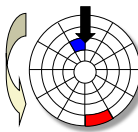
After RED read

Complete read of red

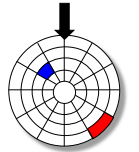
Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

29

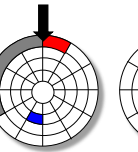
Disk Access – Service Time Components



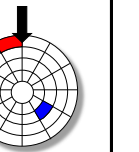
After BLUE read



Seek for RED



Rotational latency



After RED read

Data transfer      Seek      Rotational latency      Data transfer

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

30

Disk Access Time

Average time to access some target sector approximated by :

Taccess = Tavg seek + Tavg rotation + Tavg transfer

Seek time (Tavg seek)

Time to position heads over cylinder containing target sector.

Typical Tavg seek is 3—9 ms

Rotational latency (Tavg rotation)

Time waiting for first bit of target sector to pass under r/w head.

Tavg rotation = 1/2 x 1/RPMs x 60 sec/1 min

Typical Tavg rotation = 7200 RPMs

Transfer time (Tavg transfer)

Time to read the bits in the target sector.

Tavg transfer = 1/RPM x 1/(avg # sectors/track) x 60 secs/1 min.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

31

Disk Access Time Example

Given:

Rotational rate = 7,200 RPM

Average seek time = 9 ms.

Avg # sectors/track = 400.

Derived:

Tavg rotation = 1/2 x (60 secs/7200 RPM) x 1000 ms/sec = 4 ms.

Tavg transfer = 60/7200 RPM x 1/400 secs/track x 1000 ms/sec = 0.02 ms

Taccess = 9 ms + 4 ms + 0.02 ms

Important points:

Access time dominated by seek time and rotational latency.

First bit in a sector is the most expensive, the rest are free.

SRAM access time is about 4 ns/doubleword, DRAM about 60 ns

Disk is about 40,000 times slower than SRAM,

2,500 times slower than DRAM.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

32

Logical Disk Blocks

Modern disks present a simpler abstract view of the complex sector geometry:

The set of available sectors is modeled as a sequence of b-sized logical blocks (0, 1, 2, ...)

Mapping between logical blocks and actual (physical) sectors

Maintained by hardware/firmware device called disk controller.

Converts requests for logical blocks into (surface,track,sector) triples.

Allows controller to set aside spare cylinders for each zone.

Accounts for the difference in "formatted capacity" and "maximum capacity".

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

33

I/O Bus

The diagram illustrates the I/O bus architecture. At the top, a CPU chip contains a Register file, ALU, and Bus interface. Below it, Main memory is connected via a Memory bus. A central I/O bus connects to various components: a USB controller (with Mouse and Keyboard), a Graphics adapter (with Monitor), and a Disk controller (with Disk). Expansion slots for other devices like network adapters are also shown. The CPU's Bus interface connects to the I/O bus via an I/O bridge.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

34

Reading a Disk Sector (1)

This diagram shows the first step of reading a disk sector. The CPU chip (with Register file, ALU, and Bus interface) sends a command, logical block number, and destination memory address to the Disk controller via the I/O bus. The text states: "CPU initiates a disk read by writing a command, logical block number, and destination memory address to a port (address) associated with disk controller."

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

35

Reading a Disk Sector (2)

This diagram shows the second step of reading a disk sector. The Disk controller performs a direct memory access (DMA) transfer, moving data from the Disk directly into Main memory. The text states: "Disk controller reads the sector and performs a direct memory access (DMA) transfer into main memory."

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

36

### Reading a Disk Sector (3)

The diagram shows a CPU chip with a Register file, ALU, and Bus interface. It is connected to Main memory and an I/O bus. The I/O bus connects to a USB controller (Mouse, Keyboard), Graphics adapter (Monitor), and Disk controller (Disk). A red arrow indicates data flow from the Disk controller to the CPU chip.

When the DMA transfer completes, the disk controller notifies the CPU with an **interrupt** (i.e., asserts a special "interrupt" pin on the CPU)

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

37

### Solid State Disks (SSDs)

The diagram shows an I/O bus connected to a Solid State Disk (SSD). The SSD contains a Flash translation layer and Flash memory. The Flash memory is divided into blocks (Block 0, Block B-1), each containing pages (Page 0, Page 1, ..., Page P-1). Requests to read and write logical disk blocks are sent to the Flash translation layer.

- Pages: 512B to 4KB, Blocks: 32 to 128 pages
- Data read/written in units of pages.
- Page can be written only after its block has been erased
- A block wears out after about 100,000 repeated writes.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

38

### SSD Performance Characteristics

Sequential read tput	550 MB/s	Sequential write tput	470 MB/s
Random read tput	365 MB/s	Random write tput	303 MB/s
Avg seq read time	50 us	Avg seq write time	60 us

- Sequential access faster than random access
  - Common theme in the memory hierarchy
- Random writes are somewhat slower
  - Erasing a block takes a long time (~1 ms)
  - Modifying a block page requires all other pages to be copied to new block
  - In earlier SSDs, the read/write gap was much larger.

Source: Intel SSD 730 product specification.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

39

### SSD Tradeoffs vs Rotating Disks

- Advantages
  - No moving parts → faster, less power, more rugged
- Disadvantages
  - Have the potential to wear out
    - Mitigated by "wear leveling logic" in flash translation layer
    - E.g. Intel SSD 730 guarantees 128 petabyte (128 x 10<sup>15</sup> bytes) of writes before they wear out
  - In 2015, about 30 times more expensive per byte
- Applications
  - MP3 players, smart phones, laptops
  - Beginning to appear in desktops and servers

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

40

### The CPU-Memory Gap

The gap widens between DRAM, disk, and CPU speeds.

The graph shows Time (ns) on a logarithmic scale from 0.0 to 100,000,000.0 over the years 1985 to 2015. The legend includes: Disk seek time (diamond), SSD access time (square), DRAM access time (triangle), SRAM access time (circle), CPU cycle time (open square), and Effective CPU cycle time (open circle). The CPU cycle time and Effective CPU cycle time show a significant decrease over time, while the Disk seek time and SSD access time remain relatively high.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

41

### Locality to the Rescue!

The key to bridging this CPU-Memory gap is a fundamental property of computer programs known as **locality**

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

42

### Agenda

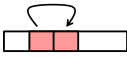
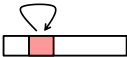
- Storage technologies and trends
- Locality of reference
- Caching in the memory hierarchy

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

43

### Locality

- **Principle of Locality:** Programs tend to use data and instructions with addresses near or equal to those they have used recently
- **Temporal locality:**
  - Recently referenced items are likely to be referenced again in the near future
- **Spatial locality:**
  - Items with nearby addresses tend to be referenced close together in time



Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

44

### Locality Example

```
sum = 0;
for (i = 0; i < n; i++)
    sum += a[i];
return sum;
```

- **Data references**
  - Reference array elements in succession (stride-1 reference pattern).
  - Reference variable `sum` each iteration.
- **Instruction references**
  - Reference instructions in sequence.
  - Cycle through loop repeatedly.

**Spatial locality**

**Temporal locality**

**Spatial locality**

**Temporal locality**

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

45

### Qualitative Estimates of Locality

- **Claim:** Being able to look at code and get a qualitative sense of its locality is a key skill for a professional programmer.
- **Question:** Does this function have good locality with respect to array `a`?

```
int sum_array_rows(int a[M][N])
{
    int i, j, sum = 0;
    for (i = 0; i < M; i++)
        for (j = 0; j < N; j++)
            sum += a[i][j];
    return sum;
}
```

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

46

### Locality Example

- **Question:** Does this function have good locality with respect to array `a`?

```
int sum_array_cols(int a[M][N])
{
    int i, j, sum = 0;
    for (j = 0; j < N; j++)
        for (i = 0; i < M; i++)
            sum += a[i][j];
    return sum;
}
```

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

47

### Locality Example

- **Question:** Can you permute the loops so that the function scans the 3-d array `a` with a stride-1 reference pattern (and thus has good spatial locality)?

```
int sum_array_3d(int a[M][N][N])
{
    int i, j, k, sum = 0;
    for (i = 0; i < M; i++)
        for (j = 0; j < N; j++)
            for (k = 0; k < N; k++)
                sum += a[k][i][j];
    return sum;
}
```

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

48



### Memory Hierarchies

- Some fundamental and enduring properties of hardware and software:
  - Fast storage technologies cost more per byte, have less capacity, and require more power (heat!).
  - The gap between CPU and main memory speed is widening.
  - Well-written programs tend to exhibit good locality.
- These fundamental properties complement each other beautifully.
- They suggest an approach for organizing memory and storage systems known as a **memory hierarchy**.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition  
49

### Agenda

- Storage technologies and trends
- Locality of reference
- Caching in the memory hierarchy

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition  
50

### Example Memory Hierarchy

Smaller, faster, and costlier (per byte) storage devices

Larger, slower, and cheaper (per byte) storage devices

L0: Regs  
CPU registers hold words retrieved from the L1 cache.

L1: L1 cache (SRAM)  
L1 cache holds cache lines retrieved from the L2 cache.

L2: L2 cache (SRAM)  
L2 cache holds cache lines retrieved from L3 cache

L3: L3 cache (SRAM)  
L3 cache holds cache lines retrieved from main memory.

L4: Main memory (DRAM)  
Main memory holds disk blocks retrieved from local disks.

L5: Local secondary storage (local disks)  
Local disks hold files retrieved from remote servers

L6: Remote secondary storage (e.g., Web servers)

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition  
51

### Caches

- Cache:** A smaller, faster storage device that acts as a staging area for a subset of the data in a larger, slower device.
- Fundamental idea of a memory hierarchy:**
  - For each  $k$ , the faster, smaller device at level  $k$  serves as a cache for the larger, slower device at level  $k+1$ .
- Why do memory hierarchies work?**
  - Because of locality, programs tend to access the data at level  $k$  more often than they access the data at level  $k+1$ .
  - Thus, the storage at level  $k+1$  can be slower, and thus larger and cheaper per bit.
- Big Idea:** The memory hierarchy creates a large pool of storage that costs as much as the cheap storage near the bottom, but that serves data to programs at the rate of the fast storage near the top.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition  
52

### General Cache Concepts

Cache

4	9	10	3
---	---	----	---

Smaller, faster, more expensive memory caches a subset of the blocks

Data is copied in block-sized transfer units

Memory

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Larger, slower, cheaper memory viewed as partitioned into "blocks"

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition  
53

### General Cache Concepts: Hit

Request: 14

Data in block  $b$  is needed

Cache

8	9	14	3
---	---	----	---

Block  $b$  is in cache: **Hit!**

Memory

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition  
54

### General Cache Concepts: Miss

Request: 12

Cache: [ 8 | 12 | 14 | 3 ]

Request: 12

Memory:

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
...	...	...	...

**Data in block *b* is needed**

**Block *b* is not in cache: Miss!**

**Block *b* is fetched from memory**

**Block *b* is stored in cache**

- **Placement policy:** determines where *b* goes
- **Replacement policy:** determines which block gets evicted (victim)

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

55

### General Caching Concepts: Types of Cache Misses

- **Cold (compulsory) miss**
  - Cold misses occur because the cache is empty.
- **Conflict miss**
  - Most caches limit blocks at level *k*+1 to a small subset (sometimes a singleton) of the block positions at level *k*.
    - E.g. Block *i* at level *k*+1 must be placed in block (*i* mod 4) at level *k*.
  - Conflict misses occur when the level *k* cache is large enough, but multiple data objects all map to the same level *k* block.
    - E.g. Referencing blocks 0, 8, 0, 8, 0, 8, ... would miss every time.
- **Capacity miss**
  - Occurs when the set of active cache blocks (**working set**) is larger than the cache.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

56

### Examples of Caching in the Mem. Hierarchy

Cache Type	What is Cached?	Where is it Cached?	Latency (cycles)	Managed By
Registers	4-8 bytes words	CPU core	0	Compiler
TLB	Address translations	On-Chip TLB	0	Hardware MMU
L1 cache	64-byte blocks	On-Chip L1	4	Hardware
L2 cache	64-byte blocks	On-Chip L2	10	Hardware
Virtual Memory	4-KB pages	Main memory	100	Hardware + OS
Buffer cache	Parts of files	Main memory	100	OS
Disk cache	Disk sectors	Disk controller	100,000	Disk firmware
Network buffer cache	Parts of files	Local disk	10,000,000	NFS client
Browser cache	Web pages	Local disk	10,000,000	Web browser
Web cache	Web pages	Remote server disks	1,000,000,000	Web proxy server

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

57

### Summary

- The speed gap between CPU, memory and mass storage continues to widen.
- Well-written programs exhibit a property called *locality*.
- Memory hierarchies based on *caching* close the gap by exploiting locality.

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

58

### Supplemental slides

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

59

### Conventional DRAM Organization

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

16 x 8 DRAM chip

rows: 0, 1, 2, 3

cols: 0, 1, 2, 3

supercell (2,1)

Internal row buffer

Memory controller

2 bits addr

8 bits data

(to/from CPU)

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

60

### 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.

16 x 8 DRAM chip

Memory controller

RAS = 2

addr

8

data

Rows

Cols

Internal row buffer

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

61

### 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.

16 x 8 DRAM chip

To CPU

Memory controller

CAS = 1

addr

8

data

Rows

Cols

Internal row buffer

supercell (2,1)

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

62

### Memory Modules

addr (row = i, col = j)

DRAM 7

DRAM 6

DRAM 5

DRAM 4

DRAM 3

DRAM 2

DRAM 1

DRAM 0

bits 56-63

bits 48-55

bits 40-47

bits 32-39

bits 24-31

bits 16-23

bits 8-15

bits 0-7

64-bit word main memory address A

Memory controller

64-bit word

□ : supercell (i,j)

64 MB memory module consisting of eight 8Mx8 DRAMs

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

63

### Enhanced DRAMs

- **Basic DRAM cell has not changed since its invention in 1966.**
  - Commercialized by Intel in 1970.
- **DRAM cores 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)**
    - Double edge clocking sends two bits per cycle per pin
    - Different types distinguished by size of small prefetch buffer:
      - DDR (2 bits), **DDR2** (4 bits), **DDR3** (8 bits)
    - By 2010, standard for most server and desktop systems
    - Intel Core i7 supports only DDR3 SDRAM

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

64

### Storage Trends

**SRAM**

Metric	1985	1990	1995	2000	2005	2010	2015	2015:1985
\$/MB	2,900	320	256	100	75	60	320	116
access (ns)	150	35	15	3	2	1.5	200	115

**DRAM**

Metric	1985	1990	1995	2000	2005	2010	2015	2015:1985
\$/MB	880	100	30	1	0.1	0.06	0.02	44,000
access (ns)	200	100	70	60	50	40	20	10
typical size (MB)	0.256	4	16	64	2,000	8,000	16,000	62,500

**Disk**

Metric	1985	1990	1995	2000	2005	2010	2015	2015:1985
\$/GB	100,000	8,000	300	10	5	0.3	0.03	3,333,333
access (ms)	75	28	10	8	5	3	3	25
typical size (GB)	0.01	0.16	1	20	160	1,500	3,000	300,000

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

65

### CPU Clock Rates

Inflection point in computer history when designers hit the "Power Wall"

	1985	1990	1995	2003	2005	2010	2015	2015:1985
CPU	80286	80386	Pentium	P-4	Core 2	Core i7(n)	Core i7(h)	
Clock rate (MHz)	6	20	150	3,300	2,000	2,500	3,000	500
Cycle time (ns)	166	50	6	0.30	0.50	0.4	0.33	500
Cores	1	1	1	1	2	4	4	4
Effective cycle time (ns)	166	50	6	0.30	0.25	0.10	0.08	2,075

(n) Nehalem processor  
(h) Haswell processor

Bryant and O'Hallaron, Computer Systems: A Programmer's Perspective, Third Edition

66