https://hao-ai-lab.github.io/dsc204a-w24/

DSC 204A: Scalable Data Systems
Winter 2024

Machine Learning Systems

Big Data

Cloud

Foundations of Data Systems

Logistics Updates

- Final Exam date (tentative): **Friday, March 22, 8 - 11 am, PT**
- Scribe notes:
  - Remember to sign-up
  - Lock down the sheet by **next Wed**
  - Might do some adjustment to balance student workload (try our best to accommodate preferences)
- Beginning of quarter survey is up, please fill the survey
- Release of the first assignment: eta 1/22, 2 weeks to finish
  - Studying Ray if you have capacity

Class Roadmap: History of Compute and Data

- ~= History of "which is the most valuable company in tech"

Google DeepMind, NVIDIA, OpenAI	Machine Learning Systems	2016 - Now
Meta, Google, NETFLIX	Big Data	2008 - 2020
CISCO, Microsoft, aws	Cloud	2000 - 2016
intel, Microsoft, Sun microsystems, IBM	Foundations of Data Systems	1980 - 2000

Where We Are

Machine Learning Systems

Big Data

Cloud

Foundations of Data Systems

1980 - 2000

Foundation of Data Systems

- Computer Organization
  - Representation of Data
  - Processors, memory, storages
- Operating System Basics
  - Processes: scheduling,
  - File systems
  - Memory management

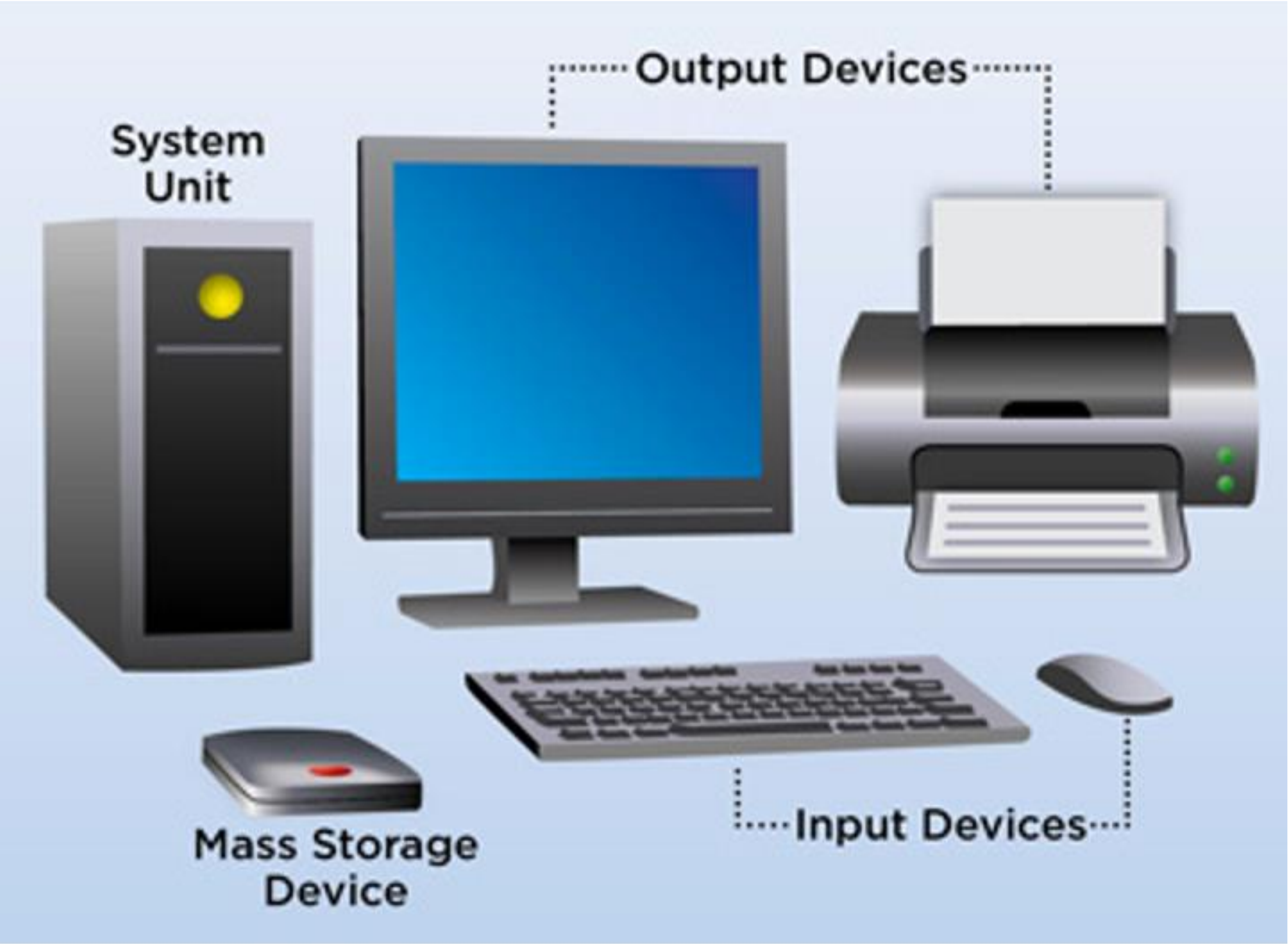
Q: What is a computer?

What is a computer?

Peter Naur

A programmable electronic device that can store, retrieve, and process digital data.

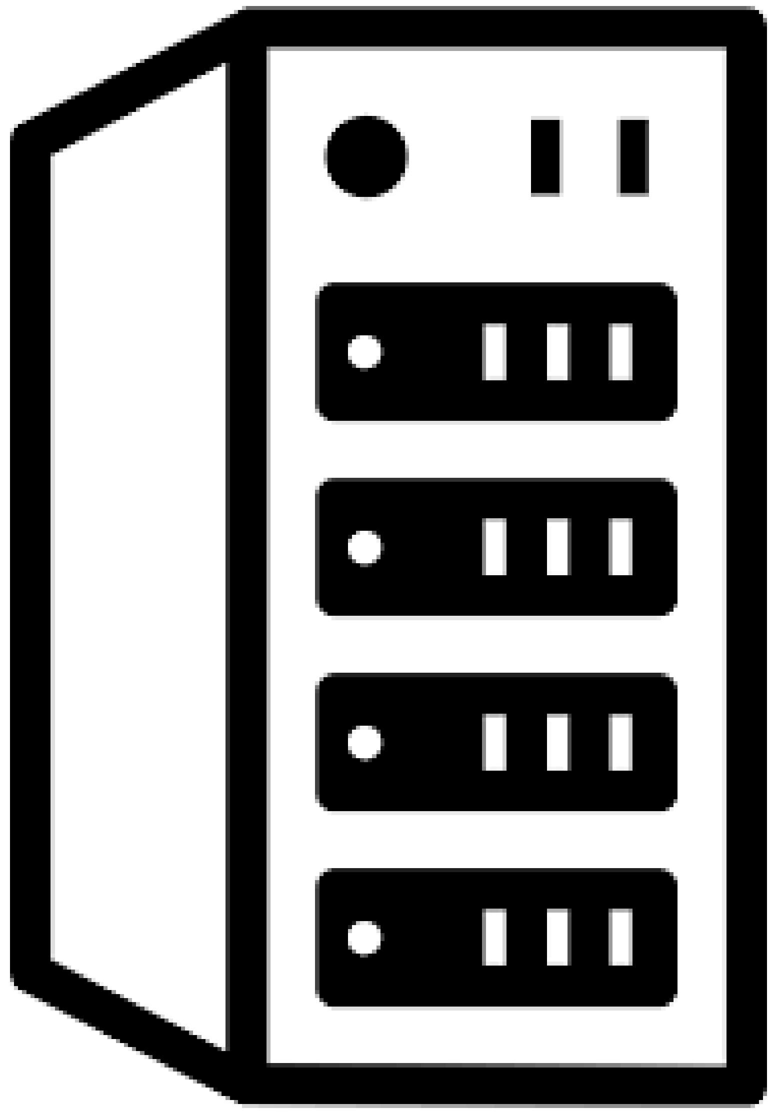
Basics of Computer Organization



- Hardware: The electronic machinery (wires, circuits, transistors, capacitors, devices, etc.)
- Software: Programs (instructions) and data

Ch. 1, 2.1-2.3, 2.12, 4.1, and 5.1-5.5 of CompOrg Book

Basics of Computer Organization



To store and retrieve data, we need:

- Disks
- Memory
- Why we need both? (we'll come back in near future)

To process data:

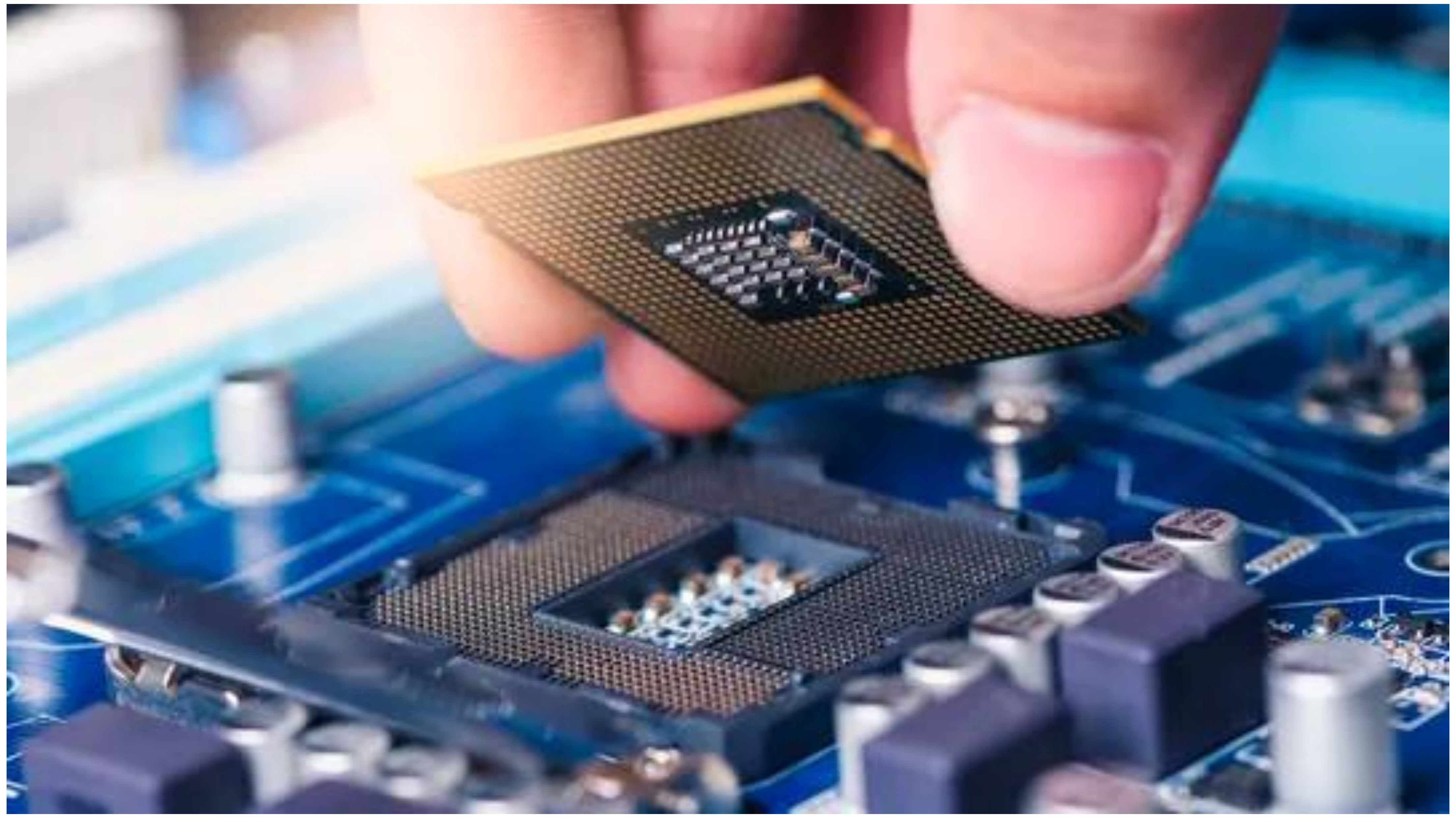
- Processors: CPU and GPU

To retrieve data from remote

- Networks

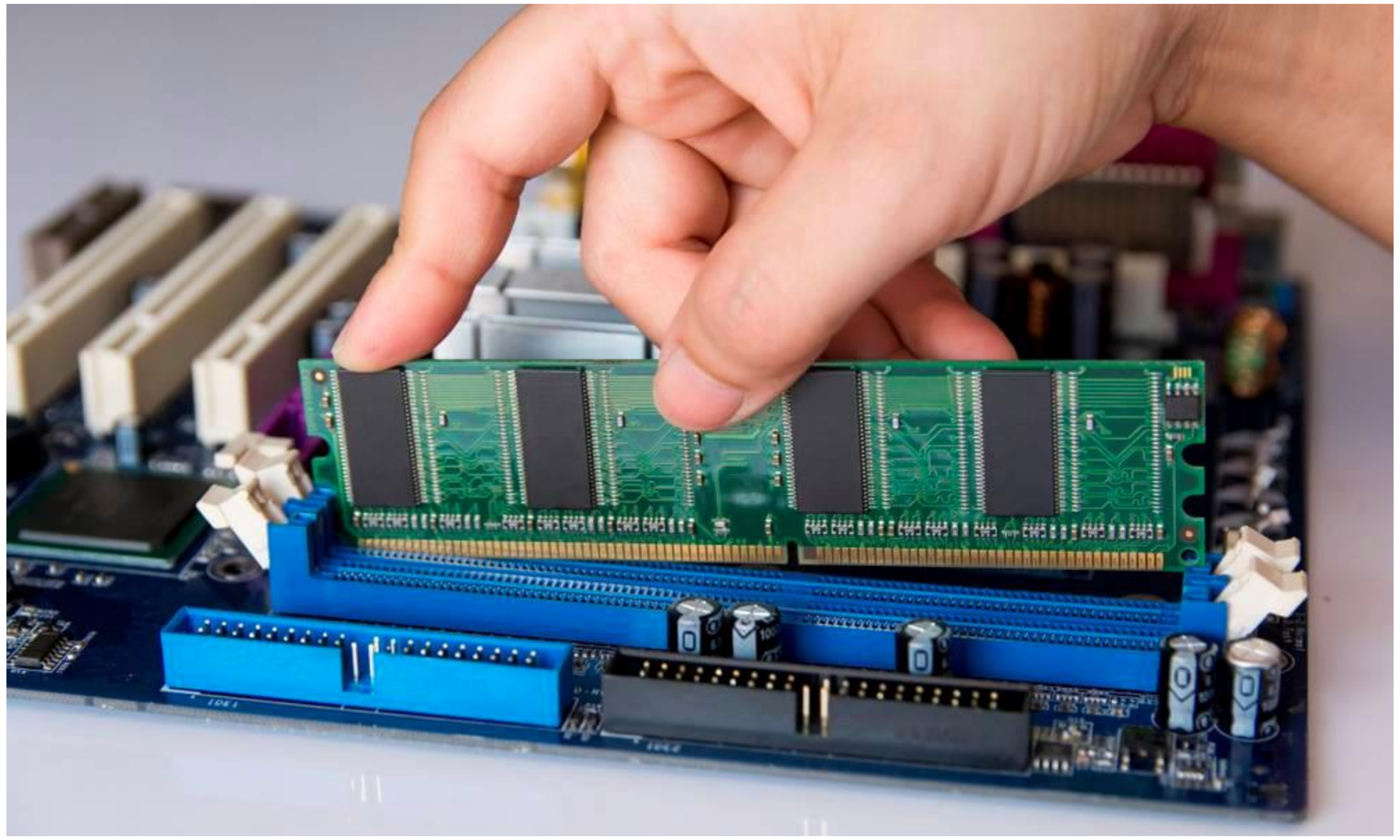
Key Parts of Computer Hardware

- Processor (CPU, GPU, etc.)
  - Hardware to orchestrate and execute instructions to manipulate data as specified by a program



Key Parts of Computer Hardware

- Main Memory (aka Dynamic Random Access Memory)
  - Hardware to store data and programs that allows very fast location/retrieval; byte-level addressing scheme

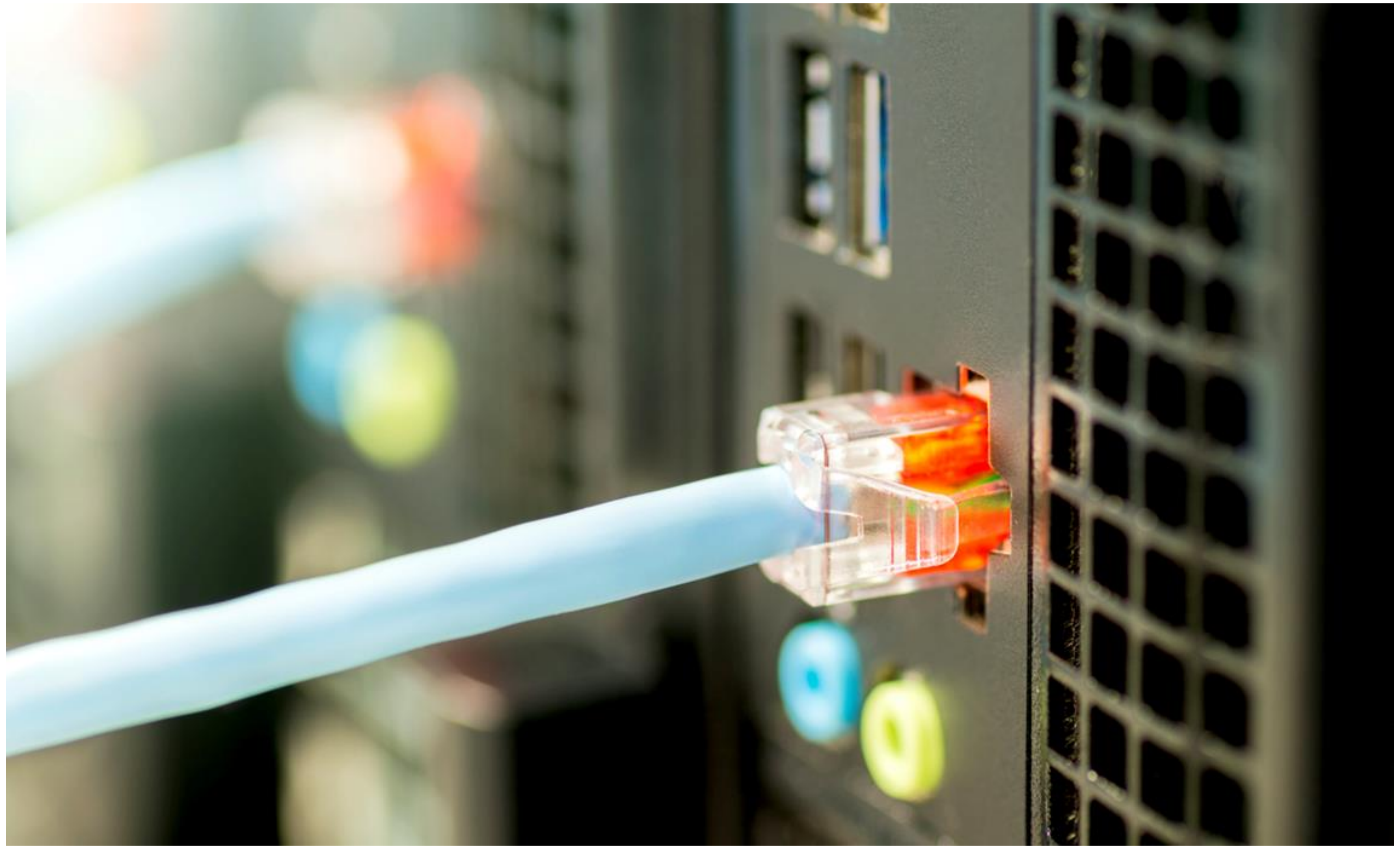


Key Parts of Computer Hardware

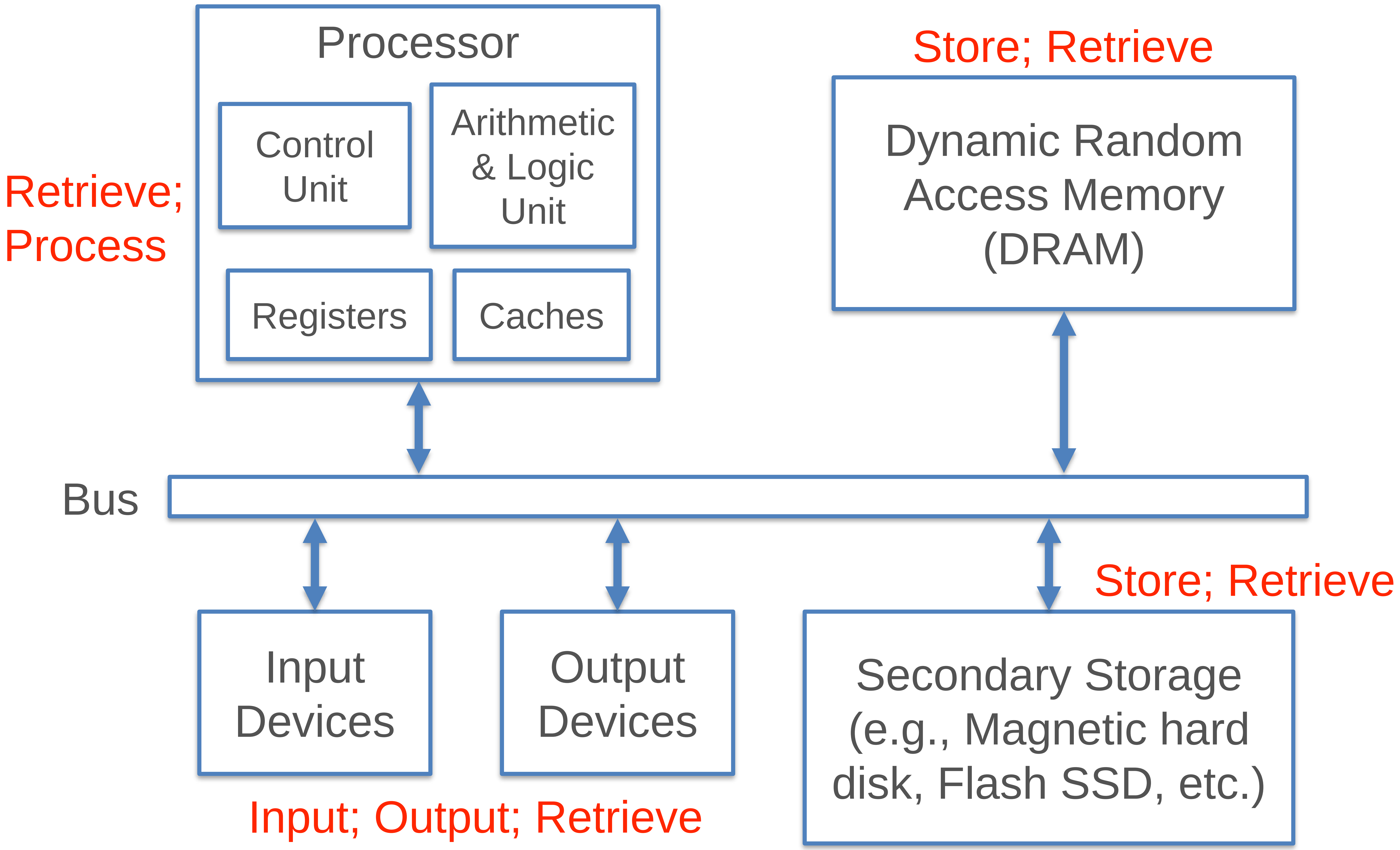
- Disk (aka secondary/persistent storage)
  - Similar to memory but persistent, slower, and higher capacity / cost ratio; various addressing schemes

Key Parts of Computer Hardware

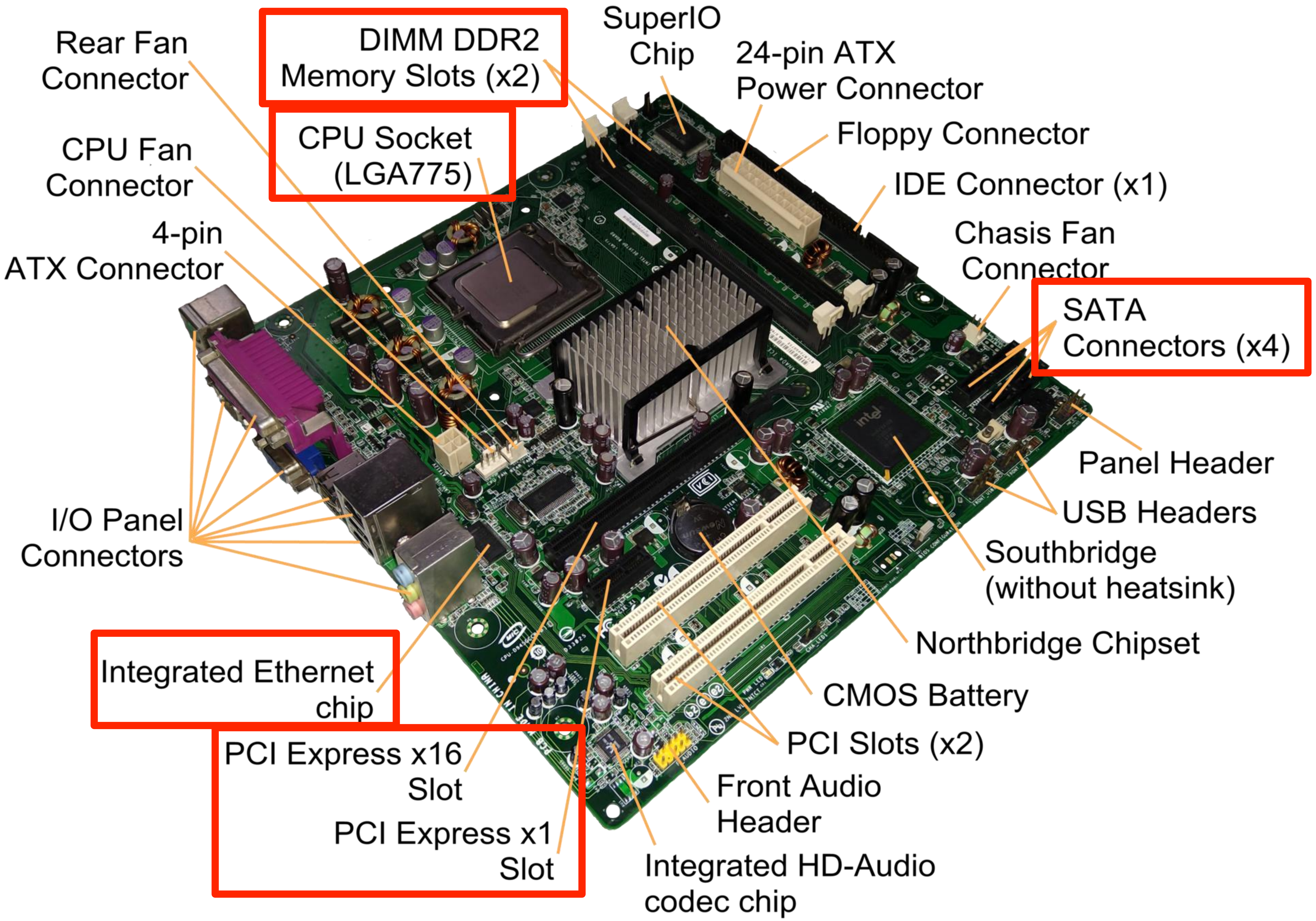
- Network interface controller (NIC)
  - Hardware to send data to / retrieve data over network of interconnected computers/devices



Abstract Computer Parts and Data

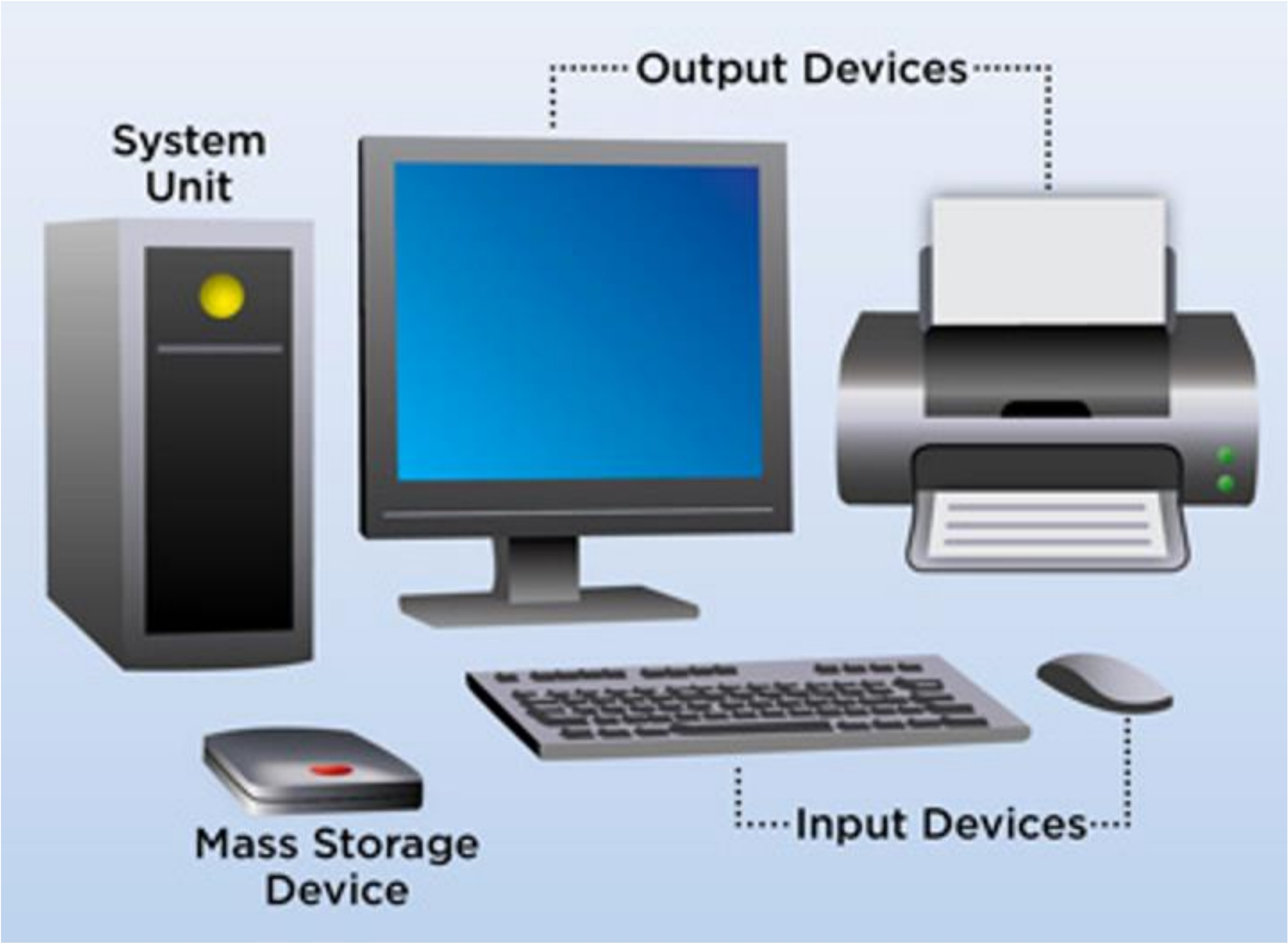


In Reality



Rear Fan Connector
DIMM DDR2 Memory Slots (x2)
SuperIO Chip
24-pin ATX Power Connector
CPU Socket (LGA775)
CPU Fan Connector
Floppy Connector
IDE Connector (x1)
4-pin ATX Connector
Chasis Fan Connector
SATA Connectors (x4)
Panel Header
USB Headers
I/O Panel Connectors
Southbridge (without heatsink)
Northbridge Chipset
Integrated Ethernet chip
CMOS Battery
PCI Express x16 Slot
PCI Slots (x2)
PCI Express x1 Slot
Front Audio Header
Integrated HD-Audio codec chip

Parts of a Computer



- Hardware: The electronic machinery (wires, circuits, transistors, capacitors, devices, etc.)
- Software: Programs (instructions) and data

https://www.webopedia.com/TERM/C/computer.html

Key Aspects of Software

- Instruction
  - A command understood by hardware; finite vocabulary for a processor: Instruction Set Architecture (ISA); bridge between hardware and software
- Program (aka code)
  - A collection of instructions for hardware to execute

Key Aspects of Software

- Programming Language (PL)
  - A human-readable formal language to write programs; at a much higher level of abstraction than ISA
- Application Programming Interface (API)
  - A set of functions ("interface") exposed by a program/set of programs for use by humans/other programs
- Data
  - Digital representation of information that is stored, processed, displayed, retrieved, or sent by a program

Main kinds of Software

- Firmware
  - Read-only programs “baked into” a device to offer basic hardware control functionalities
- Operating System (OS)
  - Collection of interrelated programs that work as an intermediary platform/service to enable application software to use hardware more effectively/easily
  - Examples: Linux, Windows, MacOS, etc.

Main kinds of Software

- Application Software
  - A program or a collection of interrelated programs to manipulate data, typically designed for human use
  - Examples: Excel, Chrome, PostgreSQL, etc.

Foundation of Data Systems

- Computer Organization
  - **Representation of Data**
  - Processors, memory, storages
- Operating System Basics
  - Processes: scheduling,
  - File systems
  - Memory management

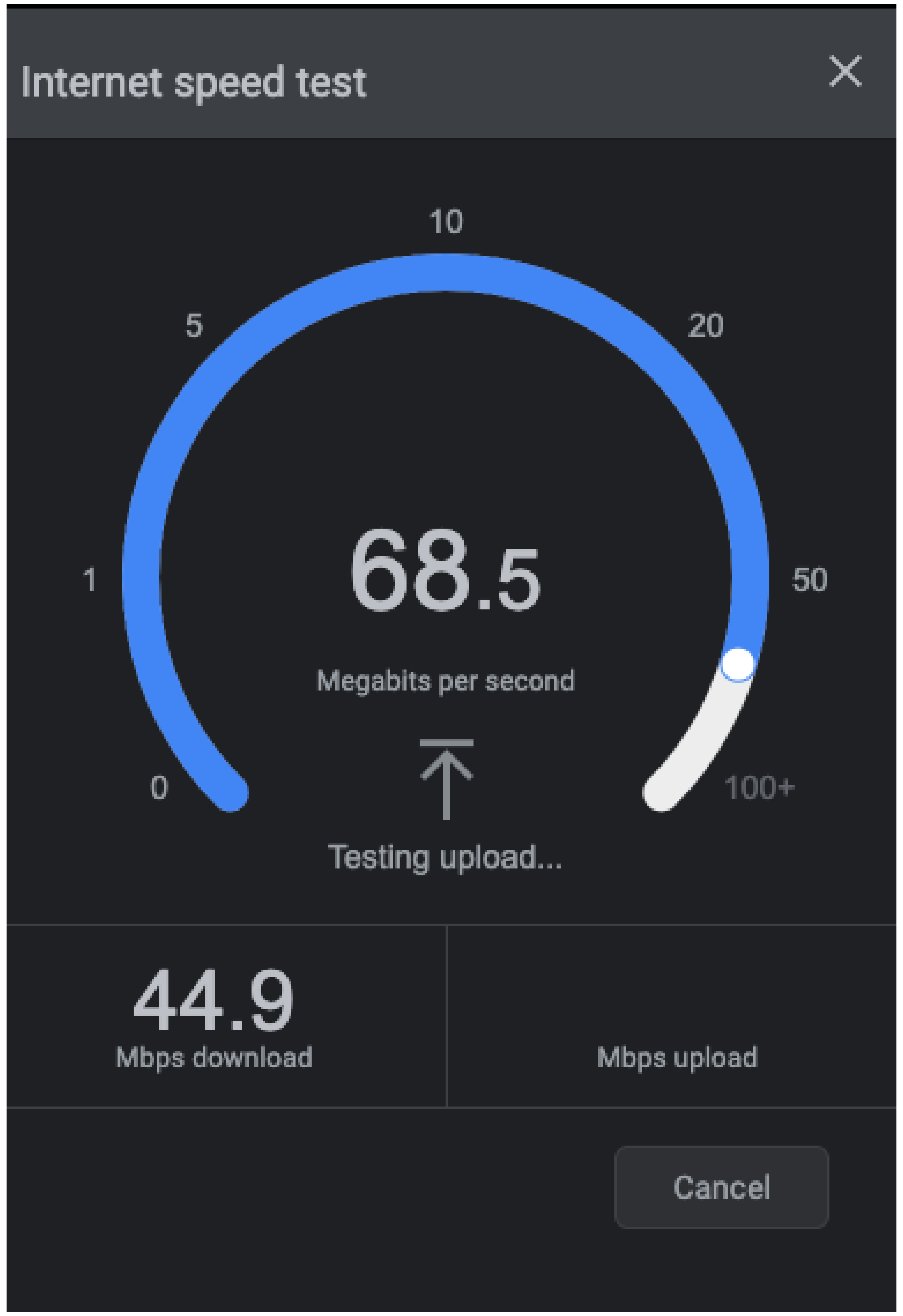
Q: How is data represented in computers?

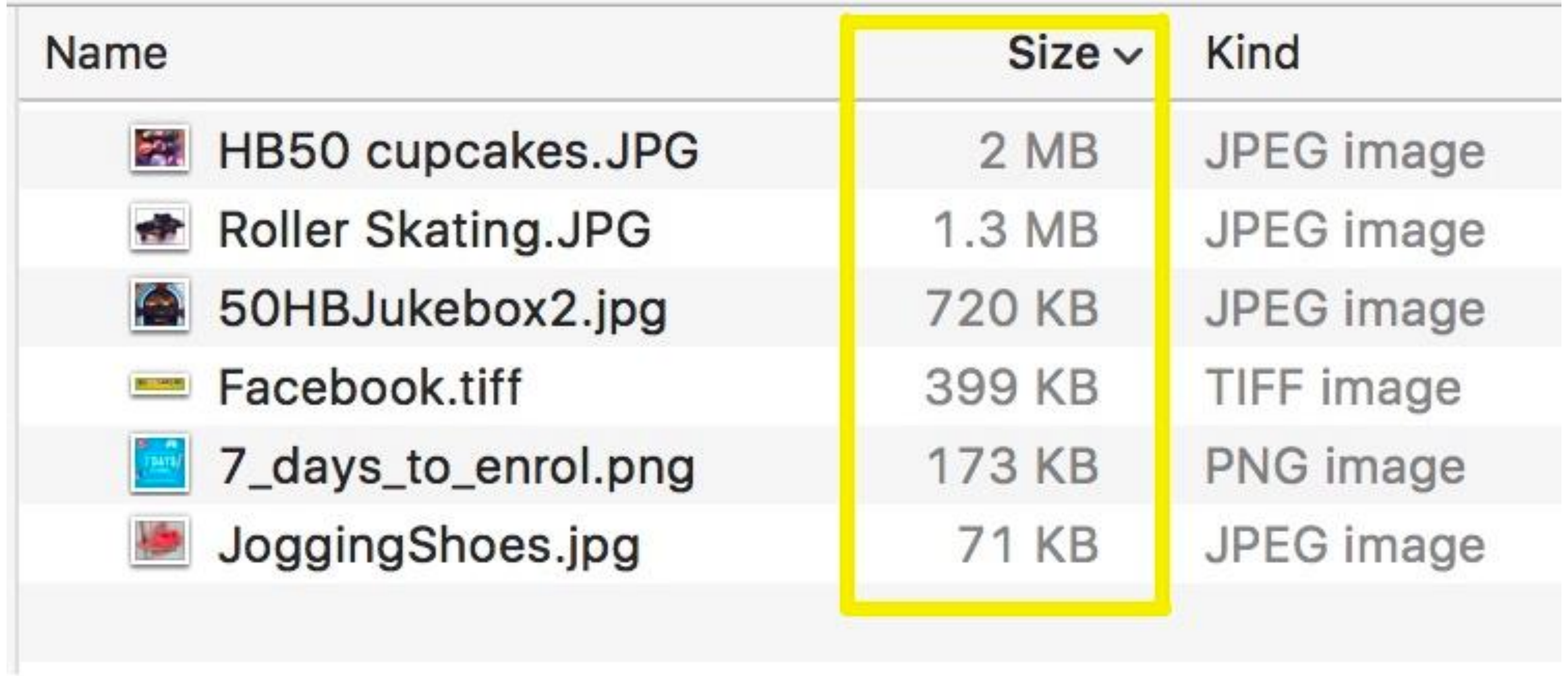
Digital Representation of Data

- Bits: All digital data are sequences of 0 & 1 (binary digits)
  - high-low/off-on electromagnetism on disk.
- Data type: First layer of abstraction to interpret a bit sequence with a human-understandable category of information; interpretation fixed by the PL
  - Example common datatypes: Boolean, Byte, Integer, “floating point” number (Float), Character, and String
- Data structure: A second layer of abstraction to *organize* multiple instances of same or varied data types as a more complex object with specified properties
  - Examples: Array, Linked list, Tuple, Graph, etc.

Count everything in binary

- Use Base 2 to represent Number
  - 0, 1, 10, 11, 100, 101, …
  - Represent 15213_{10} as $0011\ 1011\ 0110\ 1101_2$
  - Represent 1.20_{10} as $1.0011\ 0011\ 0011\ 0011\ [0011]\ldots_2$
- Represent negative numbers as …?
  - (we'll come back to this)


Internet speed test
10
5
20
1
50
0
100+
68.5
Megabits per second
Testing upload...
44.9
Mbps download
Mbps upload
Cancel


Name
Size
Kind
HB50 cupcakes.JPG
2 MB
JPEG image
Roller Skating.JPG
1.3 MB
JPEG image
50HBJukebox2.jpg
720 KB
JPEG image
Facebook.tiff
399 KB
TIFF image
7_days_to_enrol.png
173 KB
PNG image
JoggingShoes.jpg
71 KB
JPEG image

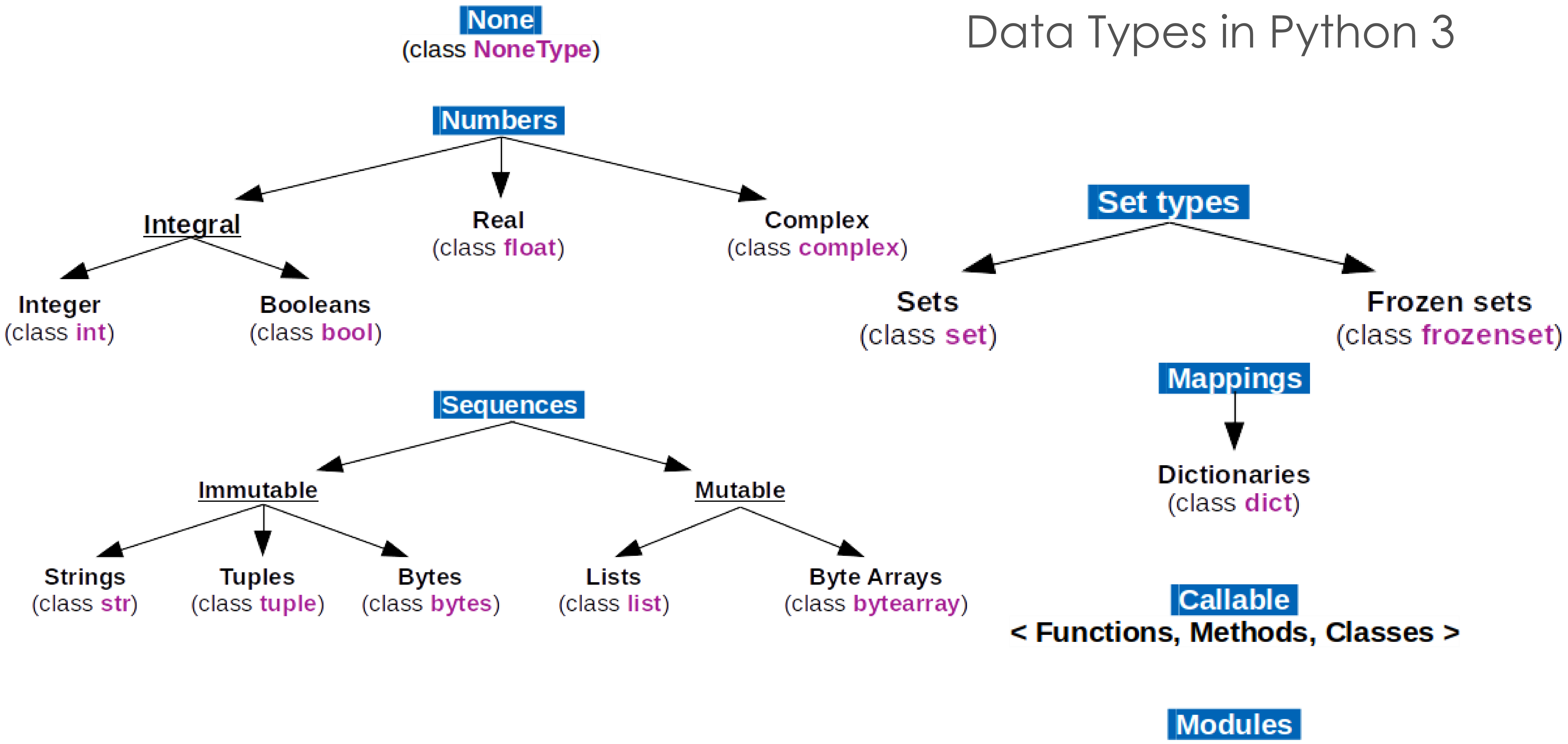
Encoding Byte Values

- Byte = 8 bits
- Why?
  - Historical Development
  - Practicality and Standardization
- A Byte (B; 8 bits) is typically the basic unit of data types
  - CPU can't address anything smaller than a byte.

Bytes -> Data types: bool, int, float, string, …

- The *size* and *interpretation* of a data type depends on PL
- Boolean:
  - Examples in data sci.: Y/N or T/F responses
  - Just 1 bit needed but actual size is almost always 1B, i.e., 7 bits are wasted!
- Integer:
  - Examples in data science: #friends, age, #likes
  - Typically 4 bytes; many variants (short, unsigned, etc.)
  - Java *int* can represent -2^{31} to $(2^{31} - 1)$; C *unsigned int* can represent 0 to $(2^{32} - 1)$;

Digital Representation of Data



Digital Representation of Data

Q: *How many unique data items can be represented by 3 bytes?*

- Given k bits, we can represent 2^k unique data items
- 3 bytes = 24 bits => 2^{24} items, i.e., 16,777,216 items
- Common approximation: 2^{10} (i.e., 1024) ~ 10^3 (i.e., 1000); recall kibibyte (KiB = 1024 B) vs kilobyte (KB = 1000 B) and so on

Q: *How many bits are needed to distinguish 97 data items?*

- For k unique items, invert the exponent to get $\log_2(k)$
- But #bits is an integer! So, we only need $\lceil \log_2(k) \rceil$
- So, we only need the next higher power of 2
- 97 ->128 = 2^7; so, 7 bits

Digital Representation of Data

Q: *How to convert from decimal to binary representation?*

- Given decimal n, if power of 2 (say, 2^k), put 1 at bit position k; if k=0, stop; else pad with trailing 0s till position 0
- If n is not power of 2, identify the power of 2 just below n (say, 2^k); #bits is then k; put 1 at position k
- Reset n as n - 2^k; return to Steps 1-2
- Fill remaining positions in between with 0s

	7	6	5	4	3	2	1	0	Position/Exponent of 2
Decimal	128	64	32	16	8	4	2	1	Power of 2
5_{10}						1	0	1	
47_{10}			1	0	1	1	1	1	
163_{10}	1	0	1	0	0	0	1	1	
16_{10}				1	0	0	0	0	

***Q:** Binary to decimal?*

Digital Representation of Data

```
void show_squares()
{
    int x;
    for (x = 5; x <= 5000000; x*=10)
            printf("x = %d x^2 = %d\n", x, x*x);
}
```

```
x = 5 x^2 = 25
x = 50 x^2 = 2500
x = 500 x^2 = 250000
x = 5000 x^2 = 25000000
x = 50000 x^2 = -1794967296
x = 500000 x^2 = 891896832
x = 5000000 x^2 = -1004630016
```

~~stack~~overflow

Two-complement: Simple Example

	-16	8	4	2	1	
10 =	0	1	0	1	0	8+2 = 10

	-16	8	4	2	1	
-10 =	1	0	1	1	0	-16+4+2 = -10

Encoding Integers

Unsigned

$$B2U(X) = \sum_{i=0}^{w-1} x_i \cdot 2^i$$

Two's Complement

$$B2T(X) = -x_{w-1} \cdot 2^{w-1} + \sum_{i=0}^{w-2} x_i \cdot 2^i$$

Sign Bit

```
short int x =  15213;
short int y = -15213;
```

Two-complement Encoding Example (Cont.)

```
x =      15213: 00111011 01101101
y =     -15213: 11000100 10010011
```

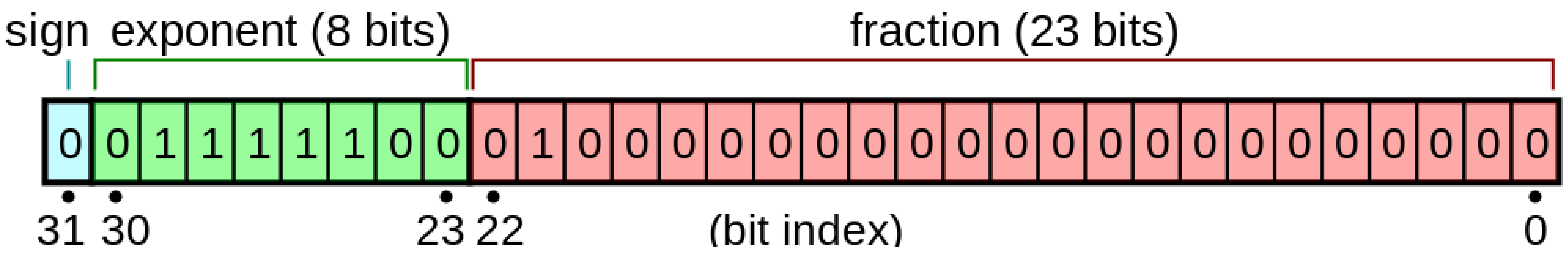
Weight	15213		-15213	
1	1	1	1	1
2	0	0	1	2
4	1	4	0	0
8	1	8	0	0
16	0	0	1	16
32	1	32	0	0
64	1	64	0	0
128	0	0	1	128
256	1	256	0	0
512	1	512	0	0
1024	0	0	1	1024
2048	1	2048	0	0
4096	1	4096	0	0
8192	1	8192	0	0
16384	0	0	1	16384
-32768	0	0	1	-32768
Sum		**15213**		**-15213**

Digital Representation of Data

- **Float**:
  - Examples in data sci.: salary, scores, model weights
  - IEEE-754 single-precision format is 4B long; double-precision format is 8B long
  - Java and C *float* is single; Python *float* is double!

Digital Representation of Data

- Float:
  - Standard IEEE format for single (aka binary32):



$$(-1)^{sign} \times 2^{exponent-127} \times (1 + \sum_{i=1}^{23} b_{23-i} 2^{-i})$$

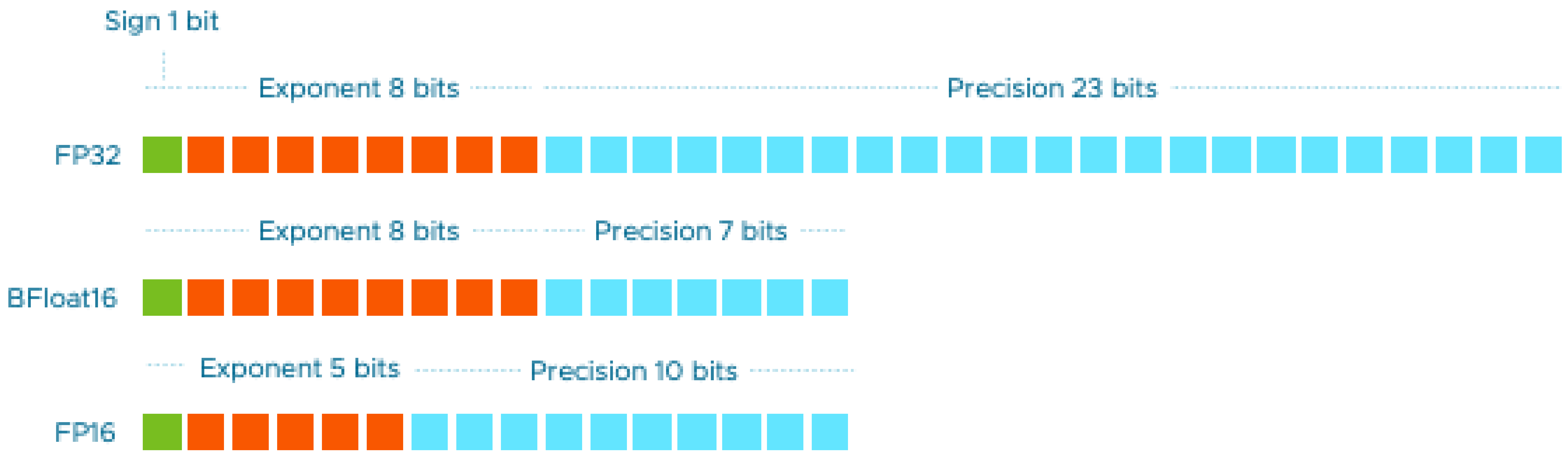
$$(-1)^0 \times 2^{124-127} \times (1 + 1 \cdot 2^{-2}) = (1/8) \times (1 + (1/4)) = 0.15625$$

Digital Representation of Data

- More float standards: double-precision (float64; 8B) and half-precision (float16; 2B); different #bits for exponent, fraction
- Float16 is now common for ***deep learning*** parameters:
  - Native support in PyTorch, TensorFlow, etc.; APIs also exist for weight quantization/rounding post training

https://docs.nvidia.com/deeplearning/sdk/mixed-precision-training/index.html

New magical float standards

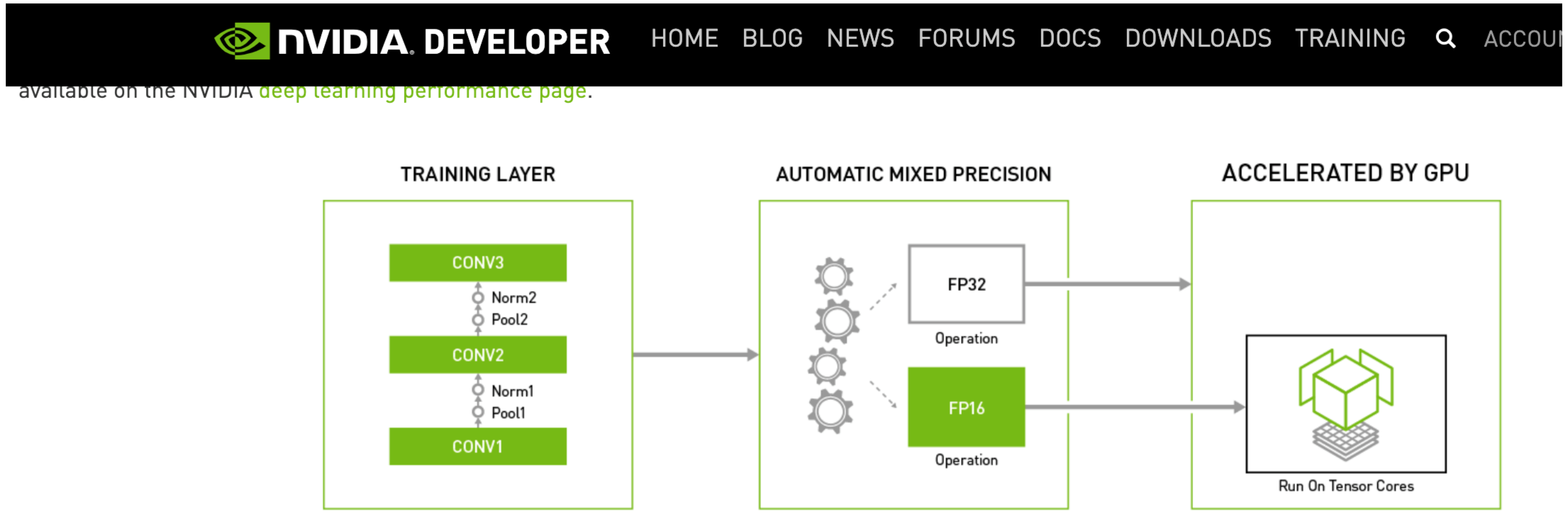


What’s the difference between bf16 and fp16?

Fp16 vs. Fp32

NVIDIA Deep Learning SDK support mixed-precision training; 2-3x speedup with similar accuracy!

Form Factor	H100 SXM
FP64	34 teraFLOPS
FP64 Tensor Core	67 teraFLOPS
FP32	67 teraFLOPS
TF32 Tensor Core	989 teraFLOPS[2]
BFLOAT16 Tensor Core	1,979 teraFLOPS[2]
FP16 Tensor Core	1,979 teraFLOPS[2]
FP8 Tensor Core	3,958 teraFLOPS[2]



Using Automatic Mixed Precision for Major Deep Learning Frameworks

Digital Representation of Data

- Representing **Character (char)** and **String**:
  - Letters, numerals, punctuations, etc.
  - A string is typically just a variable-sized array of char
  - C *char* is 1B; Java char is 2B; Python does not have a char type (use *str* or *bytes*)
  - American Standard Code for Information Interchange (*ASCII*) for encoding characters; initially 7-bit; later extended to 8-bit
    - Examples: ‘A’ is 61, ‘a’ is 97, ‘@’ is 64, ‘!’ is 33, etc.
  - *Unicode UTF-8* is now common, subsumes ASCII; 4B for ~1.1 million “code points” incl. many other language scripts, math symbols, 🤗, etc. 💻

Digital Representation of Data

- All digital objects are *collections* of basic data types (bytes, integers, floats, and characters)
  - SQL dates/timestamp: string (w/ known format)
  - ML feature vector: *array* of floats (w/ known length)
  - Neural network weights: *set* of multi-dimensional *arrays* (matrices or tensors) of floats (w/ known dimensions)
  - Graph: an *abstract data type* (ADT) with *set* of vertices (say, integers) and *set* of edges (*pair* of integers)
  - Program in PL, SQL query: string (w/ grammar)
  - Other data structures or digital objects?

Practice Qs (review next class)

Q1: How much space do I need to store GPT-3 ?

Q2: What do **exponent** and **fraction** control in float point representation?

Q3: What is the difference between BF16 and FP16?