# Layers of Abstraction in Computer System
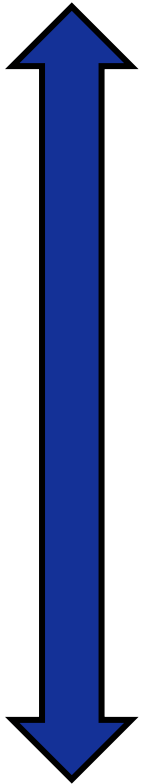
Jiaming Xu

2023.9.20

# Computer Architecture *Definition*

➢ **Bridge application and technology**

Application

⬆⬇

**Gap too large** to bridge in one step
➜ **Computer Architecture:** develop abstraction and implementation layers to execute information processing application efficiently using available fabrication technology

Technology

CHASELab *Slides from Jie Zhang, Computer Architecture*

# Why Abstraction

- Better develop
  - The Moore's law
  - Years of development
    - More layers, the quicker



Moore's Law: The number of transistors on microchips doubles every two years

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.
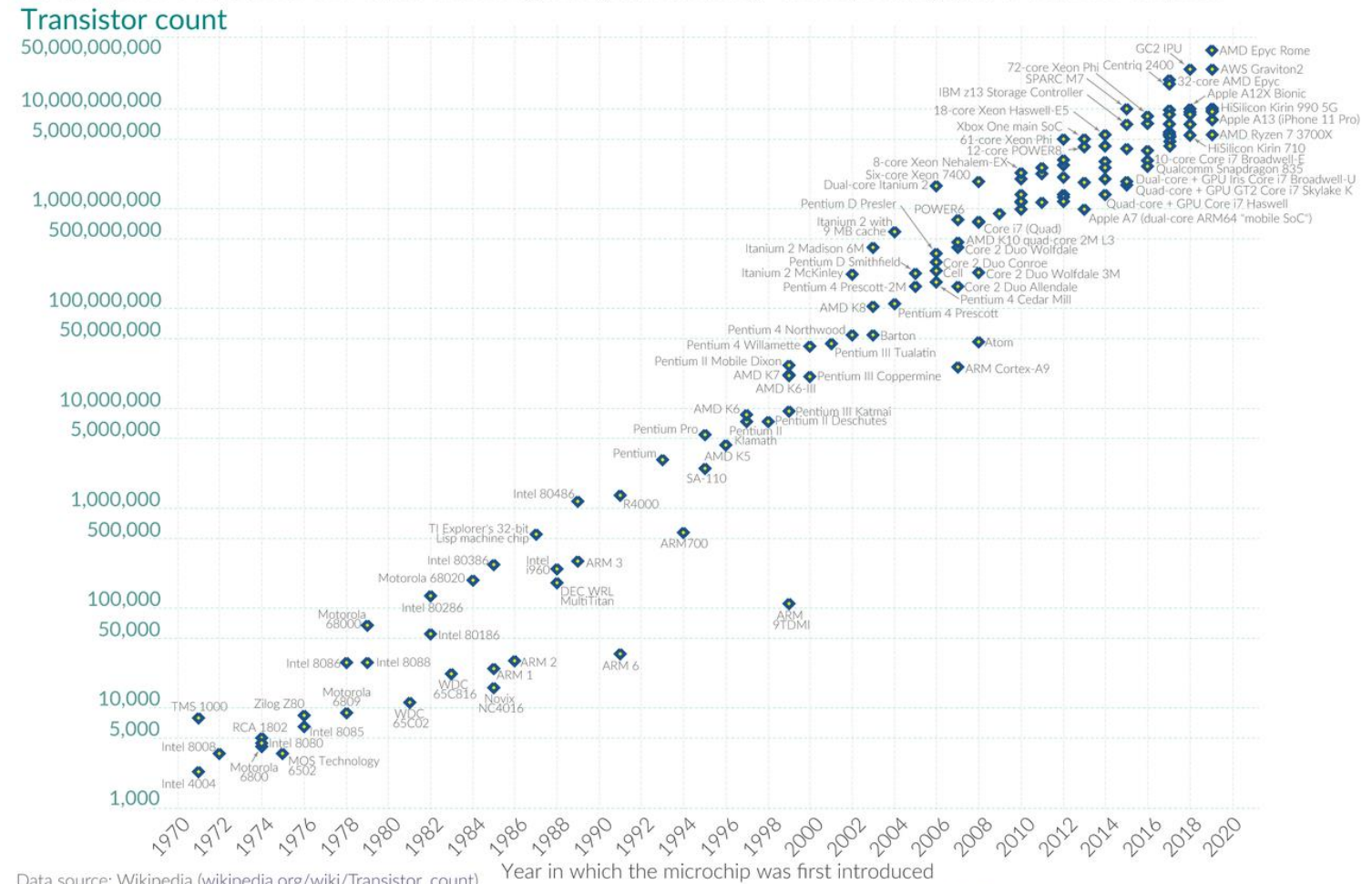
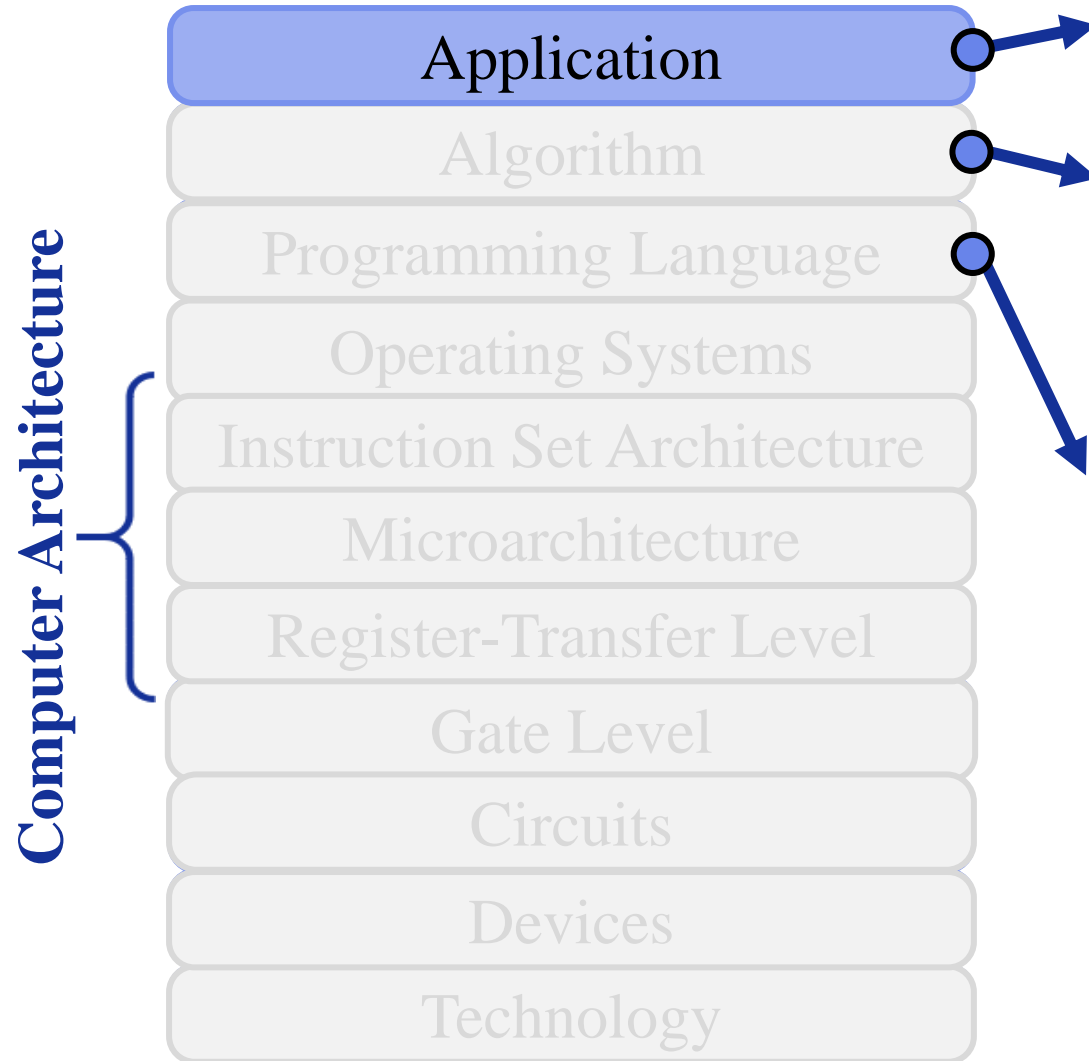Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
OurWorldinData.org – Research and data to make progress against the world's largest problems.

Licensed under CC-BY by the authors Hannah Ritchie and Max Roser.

# Why Abstraction

- Better use
  - abstracts out low-level implementation details


- Good or not?


- What is the bottleneck of today's software?
  - Memory? No
  - Bandwidth? May not
  - ...
  - Energy inefficiency
    - 3GW for Google's data center
    - 800MW for Meta's data center

# Computer System Stack



**Computer Architecture**

| Application |
|---|
| Algorithm |
| Programming Language |
| Operating Systems |
| Instruction Set Architecture |
| Microarchitecture |
| Register-Transfer Level |
| Gate Level |
| Circuits |
| Devices |
| Technology |

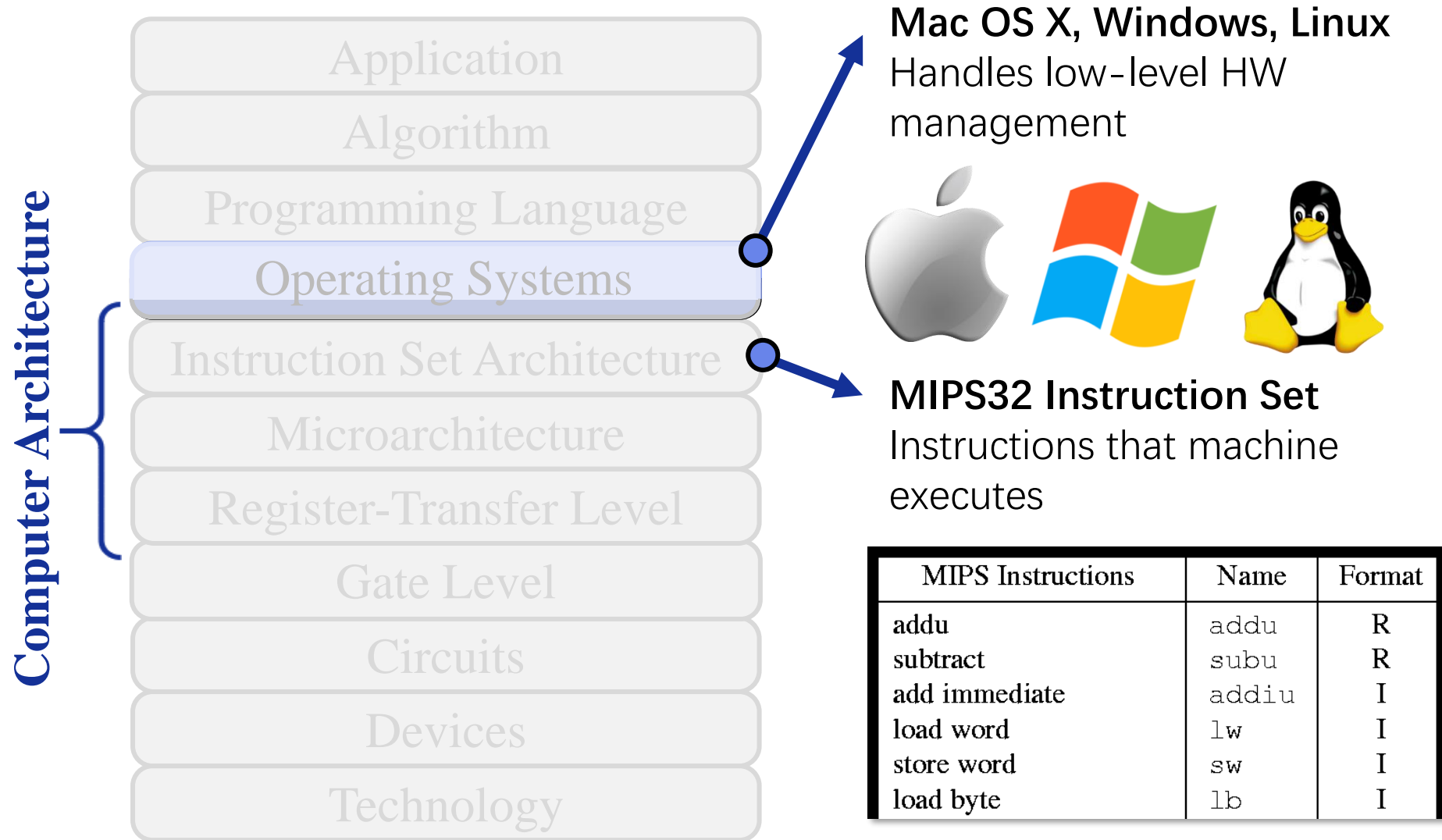**Ex) Sort an array of numbers**
2,6,3,8,4,5 → 2,3,4,5,6,8

**Out-of-place selection sort algorithm**
1. Find *min* number in array
2. Move *min* number into output array
3. Repeat steps 1 & 2 until finished

**C implementation of selection sort**

```
 1  void sort( int b[], int a[], int n ) {
 2      for ( int idx, k = 0; k < n; k++ ) {
 3          int min = 100;
 4          for ( int i = 0; i < n; i++ ) {
 5              if ( a[i] < min ) {
 6                  min = a[i];
 7                  idx = i;
 8              }
 9          }
10          b[k] = min;
11          a[idx] = 100;
12      }
13  }
```

*Slides from Jie Zhang, Computer Architecture*

# Computer System Stack

**Computer Architecture** {.vertical}

| Application |
| Algorithm |
| Programming Language |
| Operating Systems |
| Instruction Set Architecture |
| Microarchitecture |
| Register-Transfer Level |
| Gate Level |
| Circuits |
| Devices |
| Technology |

**Mac OS X, Windows, Linux**
Handles low-level HW management

**MIPS32 Instruction Set**
Instructions that machine executes

| MIPS Instructions | Name | Format |
|---|---|---|
| addu | addu | R |
| subtract | subu | R |
| add immediate | addiu | I |
| load word | lw | I |
| store word | sw | I |
| load byte | lb | I |

# Computer System Stack

**Computer Architecture**

| Application |
|:---:|
| Algorithm |
| Programming Language |
| Operating Systems |
| Instruction Set Architecture |
| Microarchitecture |
| **Register-Transfer Level** |
| Gate Level |
| Circuits |
| Devices |
| Technology |

How data flows through system

Boolean logic gates and functions

Combining devices to do useful work

Transistors and wires

Silicon process technology

# Logic, State, and Interconnect

**Computer Architecture**

Application

Algorithm

Programming Language

Operating Systems

Instruction Set Architecture

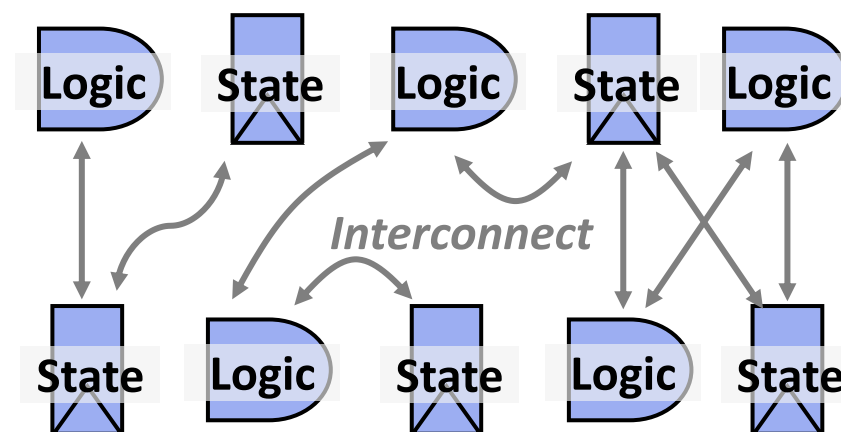Microarchitecture

Register-Transfer Level

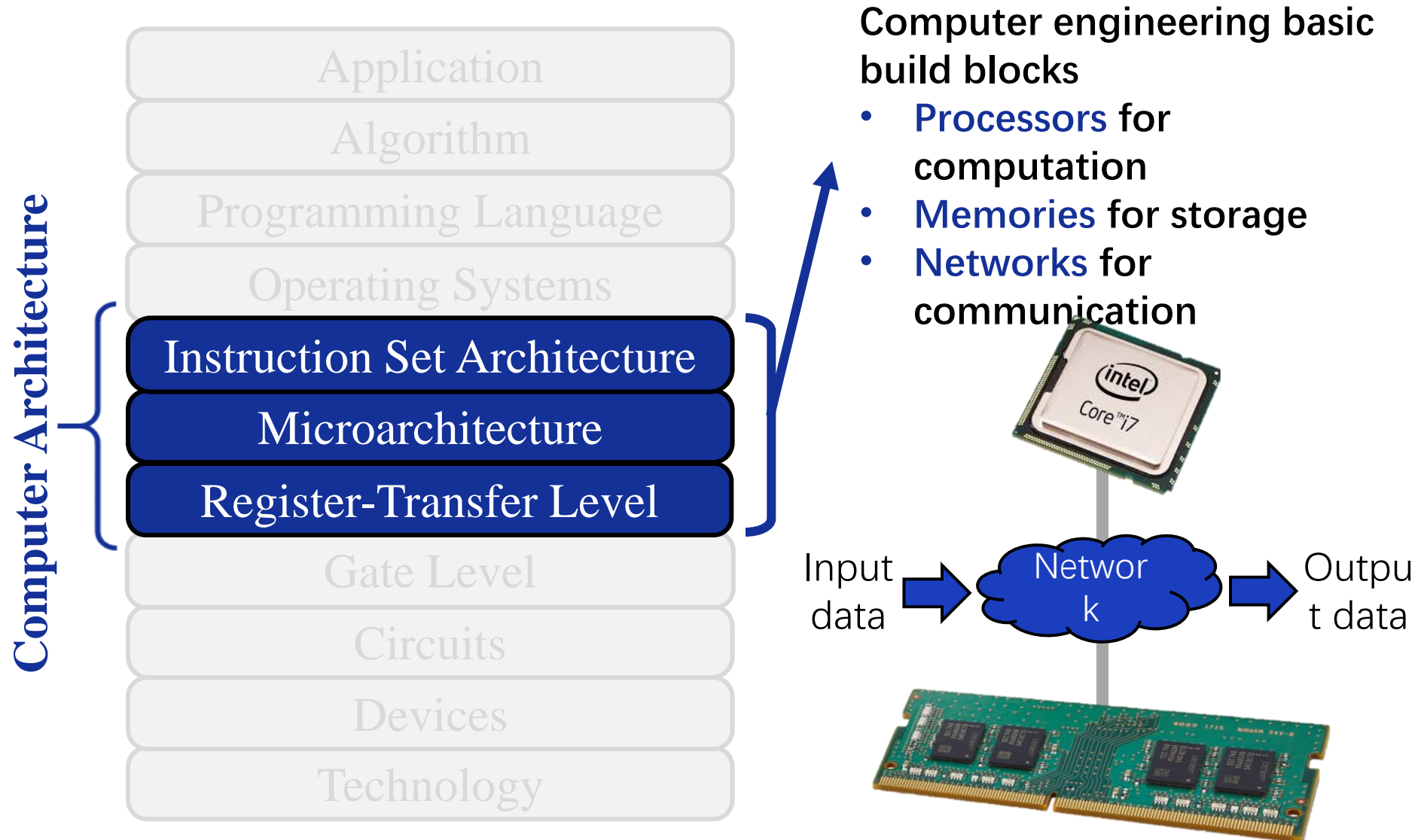**Gate Level**

Circuits

Devices

Technology
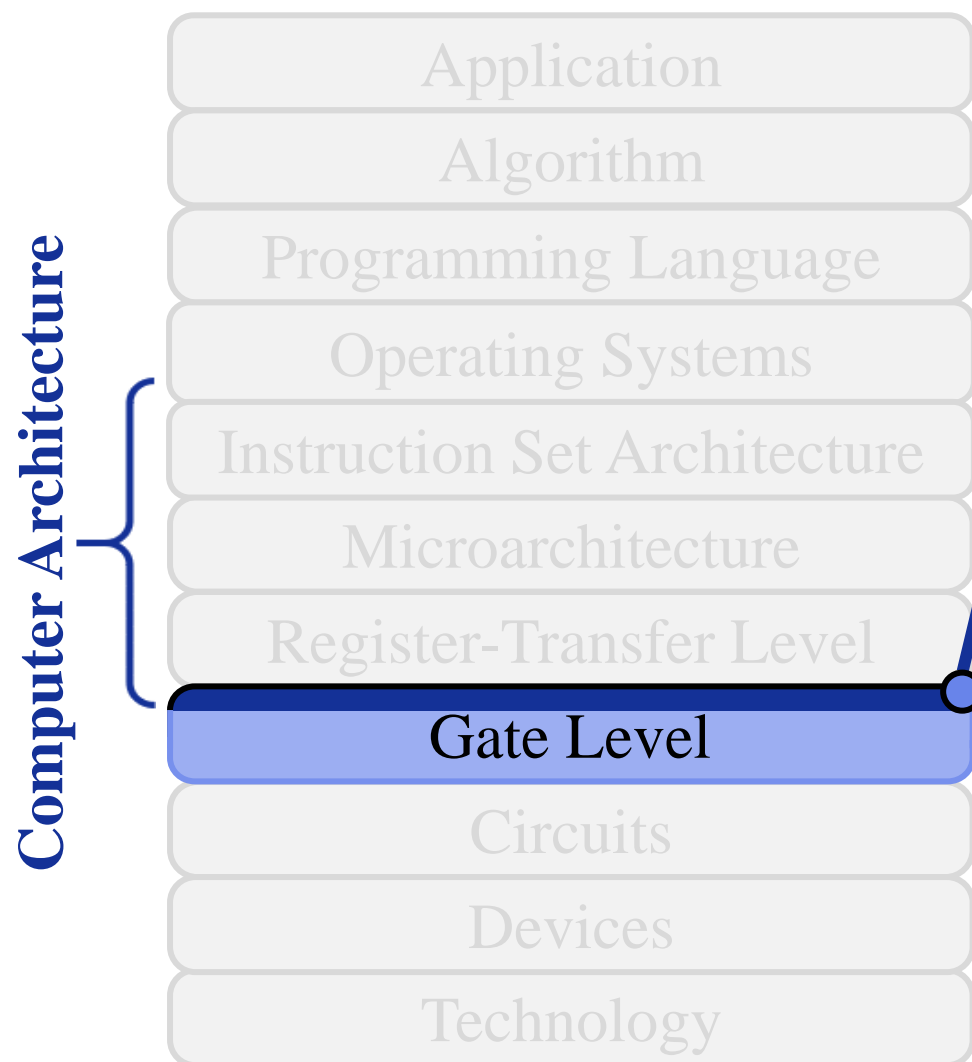
Digital systems are implemented with three basic building blocks
- **Logic** to process data
- **State** to store data
- **Interconnect** to move data

**Logic**  **State**  **Logic**  **State**  **Logic**

*Interconnect*

**State**  **Logic**  **State**  **Logic**  **State**

# General-Purpose Computing:

Computer Architecture

- Application
- Algorithm
- Programming Language
- Operating Systems
- **Instruction Set Architecture**
- **Microarchitecture**
- **Register-Transfer Level**
- Gate Level
- Circuits
- Devices
- Technology

Computer engineering basic build blocks

- **Processors** for computation
- **Memories** for storage
- **Networks** for communication
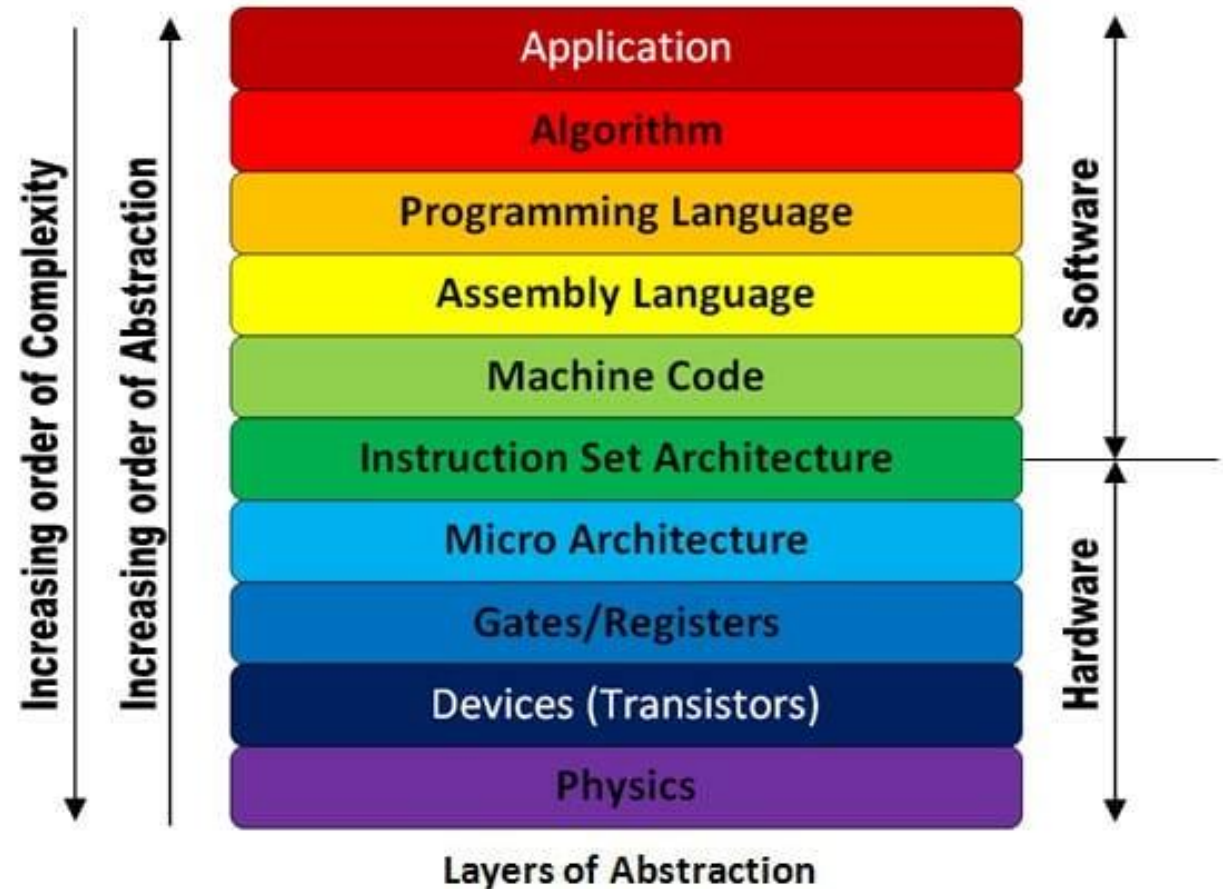
Input data → Network → Output data

# Other Version of Abstractions
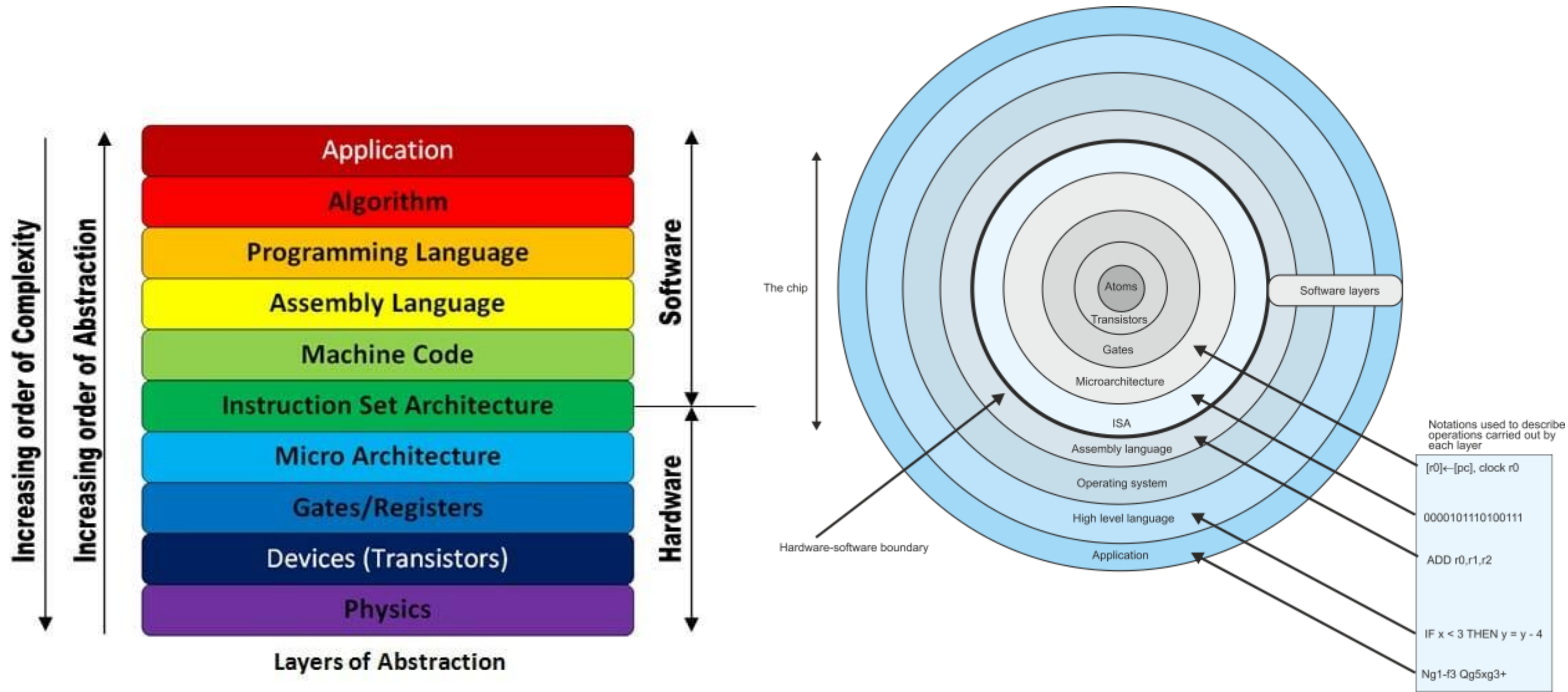
# Development of CODE

- Electrical signal (on/off)
- Machine code (0/1)
- Assembly language
  - Some notations help to communicate with machine
- High-level language
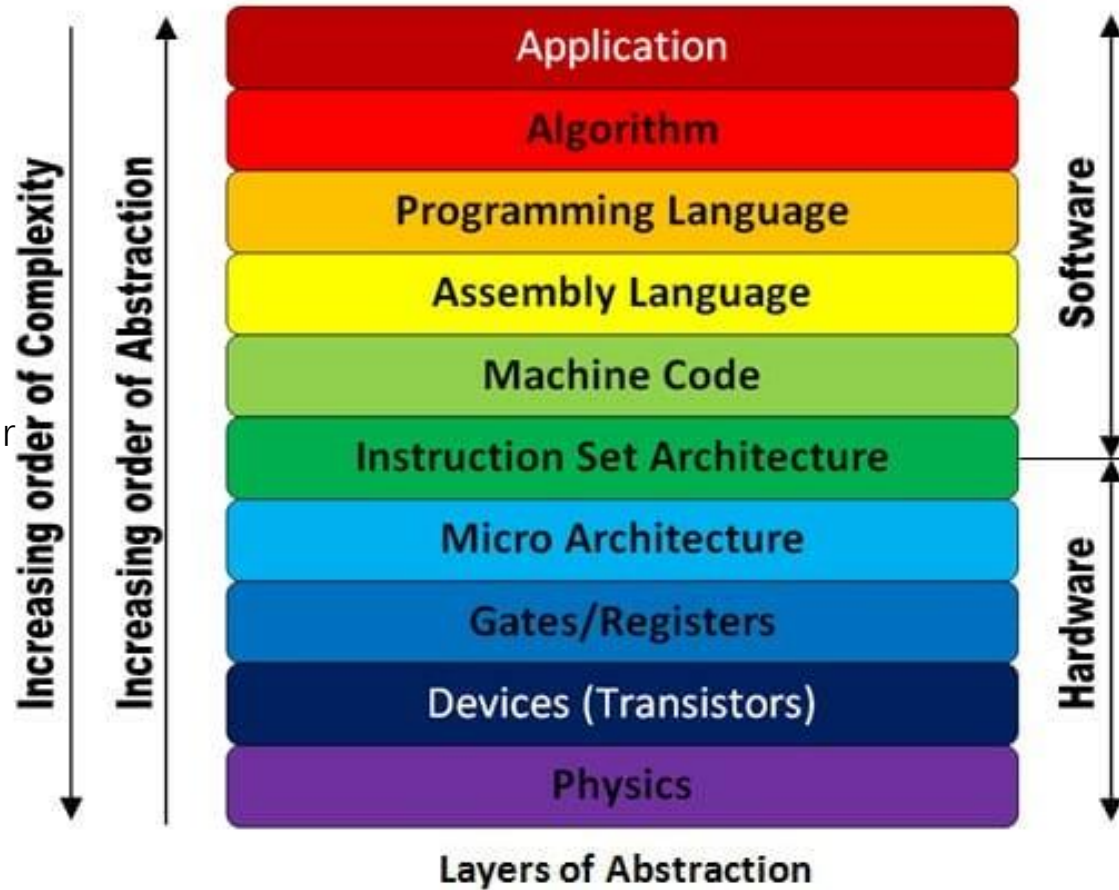  - More abstractive
  - More human-mindset



**Layers of Abstraction**

# Examples of CODE

- C/C++ (.c/.cpp, .h/.hpp)

GCC
↓ Compiler
- Assembly language (.s)
↓ Assembler
- Machine Code (.o, .so, .a)
↓ Linker

CLANG (LLVM)
↓ Compiler
- LLVM IR
↓ LLVM Optimizer
- LLVM Backend
↓ Assembler
↓ Linker

**Increasing order of Complexity**

**Increasing order of Abstraction**

| Application |
| --- |
| Algorithm |
| Programming Language |
| Assembly Language |
| Machine Code |
| Instruction Set Architecture |
| Micro Architecture |
| Gates/Registers |
| Devices (Transistors) |
| Physics |

**Software**

**Hardware**

**Layers of Abstraction**

- Java

JVM
↓ Compiler
- Java Bytecode
(Java Virtual Machine)
↓ Assembler
↓ Linker
- Machine Code

JIT
↓ Compiler
- Java Bytecode
↓ Just In Time compiler
- Machine code
- (Java Virtual Machine)
↓ Linker
- Machine Code

# Thanks & Advertisement

# Who Am I?

- **Jie Zhang**
  - Ph.D. advisor, Boya Young Scholar, NSFC Excellent Overseas Young Scientists Fund
  - Assistant Professor, Peking University
  - KAIST, South Korea, Postdoctoral Researcher in Computer Science
  - Yonsei University, South Korea, Ph.D. in Computer Science
  - Web: https://jiezhang-camel.github.io/

CHASELab

# CHASE Lab

- Research:
  - Computer architecture
  - Storage and memory systems
- Members:
  - Shushu Yi (Ph.D., graduated from Nanjing University)
  - Xiurui Pan (intern @ Tsinghua University)
  - Li Peng (intern @ HUST)
  - Yuda An, Zeyu Li (interns @ Peking University)
- Research collaborators

**We are hiring**

If you are interested in CA, storage or research
please send your information to
**jiez@pku.edu.cn**

CHASELab

# Reference

- Computer Organization And Design: The Hardware/Software Interface (David a. Patterson, John L. Hennessy) Chapter 1, 2
- [Moore's law – Wikipedia](#)
- [Computer Hierarchy (alanclements.org)](#)
- [Understanding the Layers of a Computer System - Secplicity - Security Simplified](#)