CSC 3210 Computer organization and programming

Chunlan Gao

Chapter1

Computer Abstractions and Technology

Last time.....

Levels of Program code

High-level language program (in C)

```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```

↓

Compiler

↓

Assembly language program (for RISC-V)

```
swap:
      slli x6, x11, 3
      add  x6, x10, x6
      ld   x5, 0(x6)
      ld   x7, 8(x6)
      sd   x7, 0(x6)
      sd   x5, 8(x6)
      jalr x0, 0(x1)
```

↓

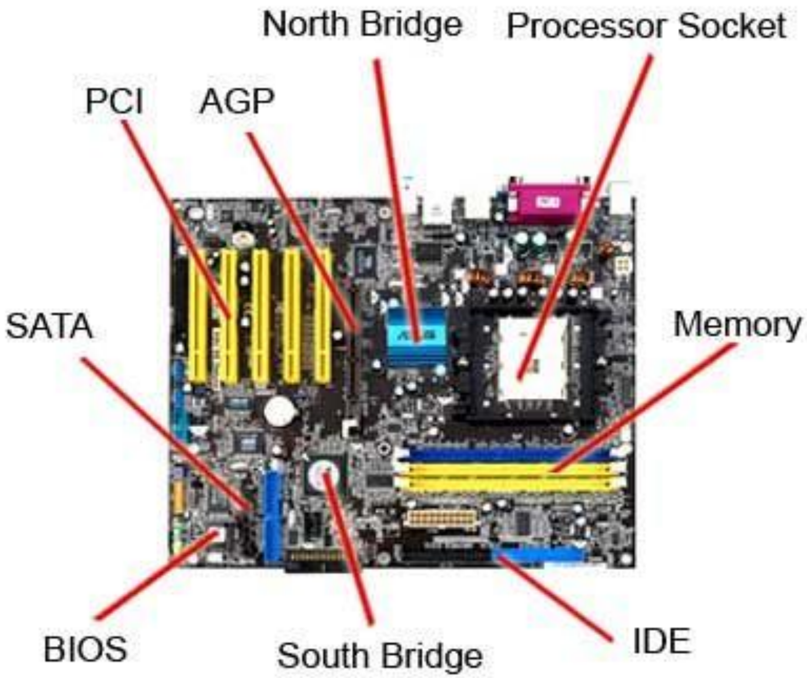
Assembler

↓

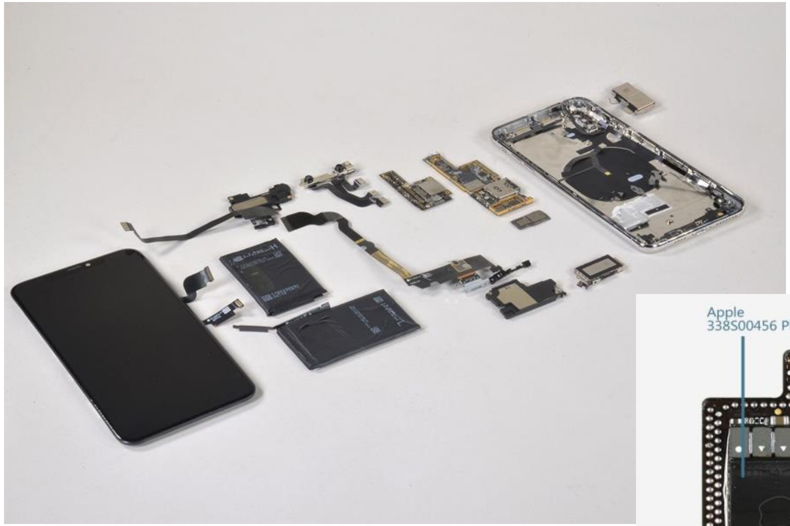
Binary machine language program (for RISC-V)

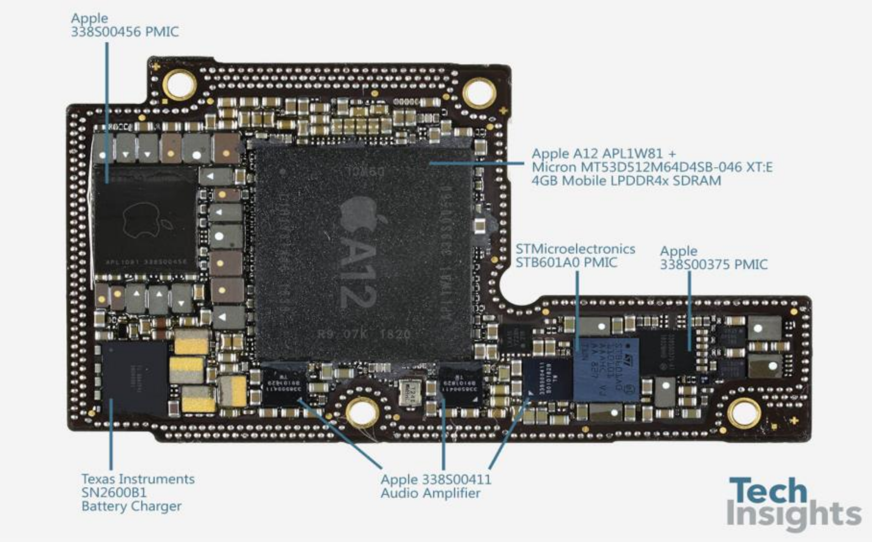
```
00000000001101011001001100010011
00000000011001010000001100110011
00000000000000110011001010000011
00000000100000110011001110000011
00000000011100110011000000100011
00000000010100110011010000100011
00000000000000001000000001100111
```

In a pc?


North Bridge
Processor Socket
PCI
AGP
SATA
Memory
BIOS
South Bridge
IDE

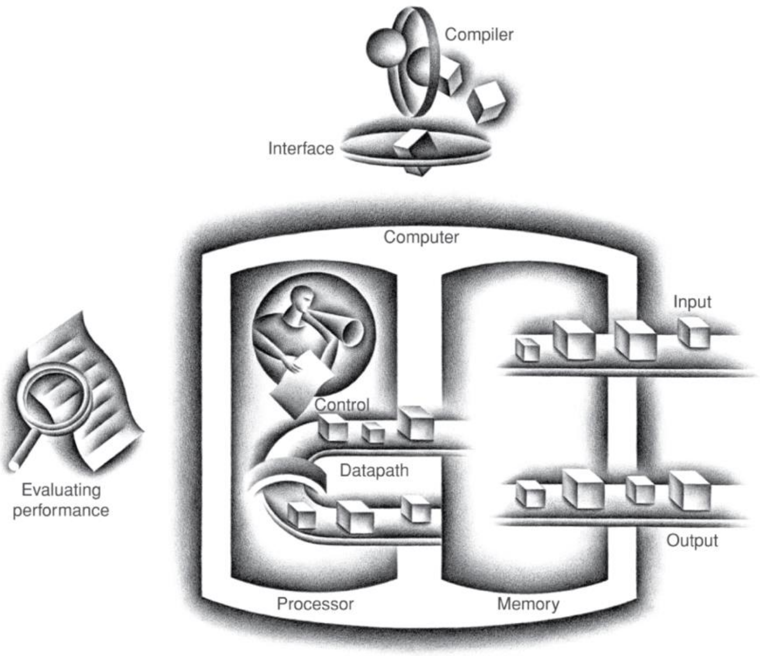
Teardown of a smartphone





Components of a Computer

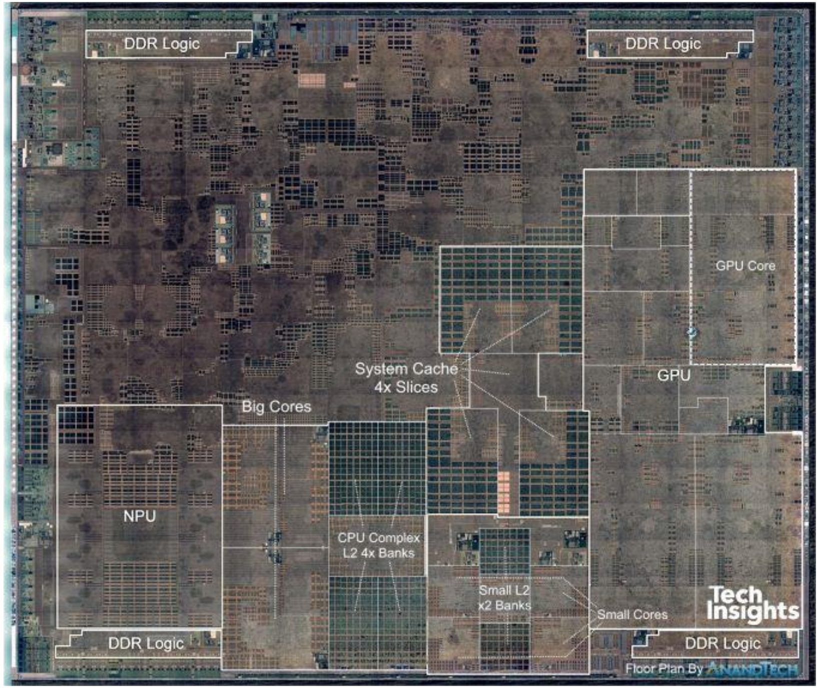
All kinds of computer have same components： Processor, memory and I/O



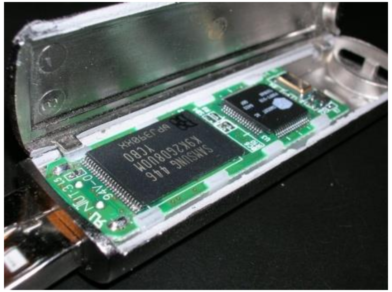
What’s inside the processor(CPU)

- Datapath: performs operations on data
- Control: sequences datapath, memory, ...
- Cache memory
  - Small fast SRAM memory for immediate access to data

A12 microprocessor



A Safe Place for Data



- Volatile main memory
  - Loses instructions and data when power off
- Non-volatile secondary memory
  - Magnetic disk
  - Flash memory
  - Optical disk (CDROM, DVD)

Electronics technology trends

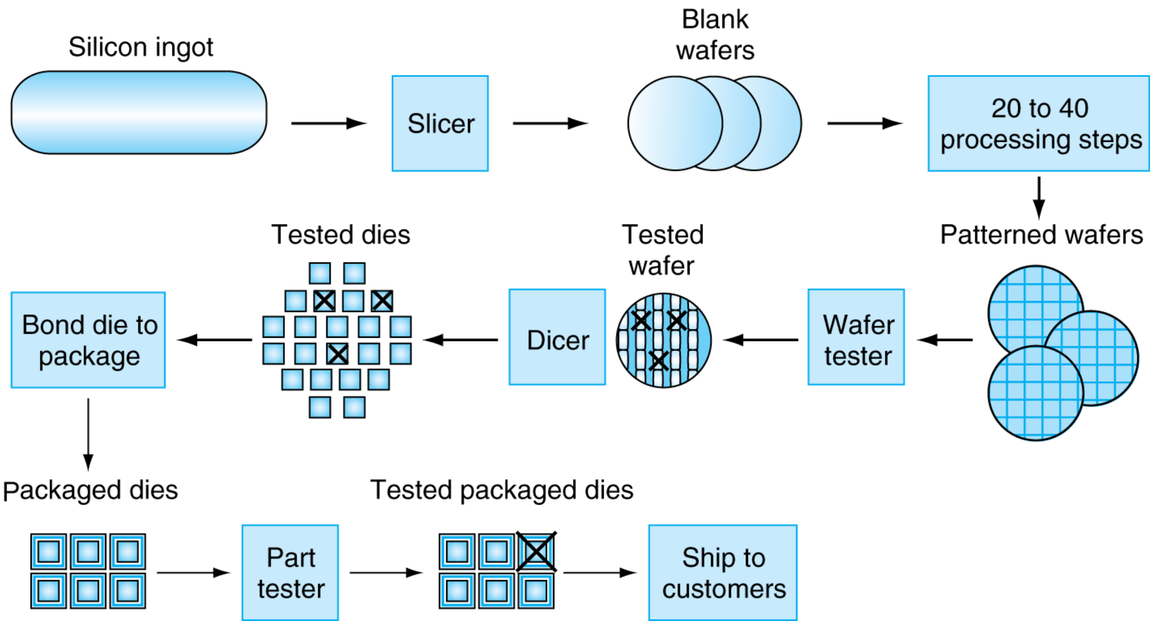
- Electronics technology continues to evolve
  - Increased capacity and performance
  - Reduced cost

Year	Technology	Relative performance/cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit (IC)	900
1995	Very large scale IC (VLSI)	2,400,000
2013	Ultra large scale IC	250,000,000,000

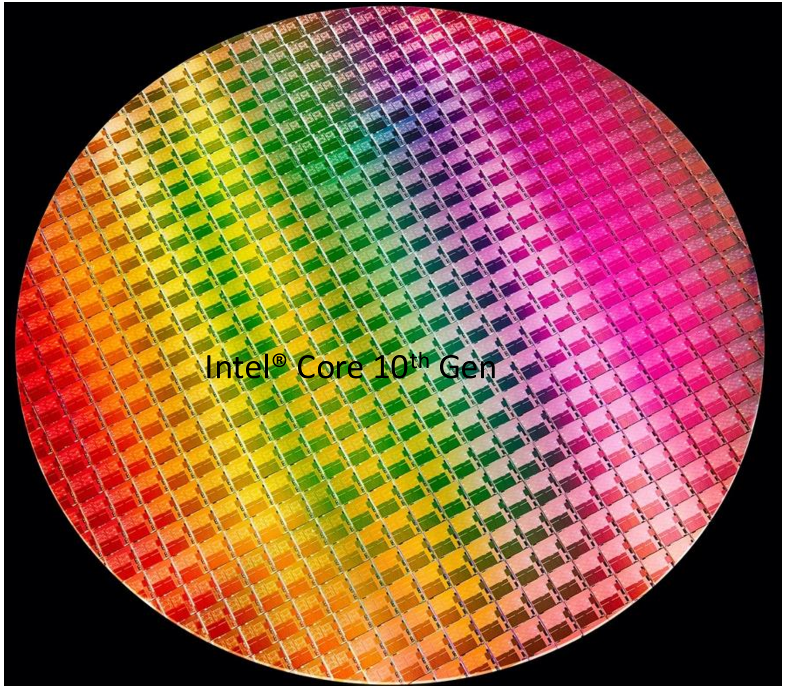
Semiconductor Technology

- Silicon: semiconductor
- Add materials to transform properties:
  - Conductors
  - Insulators
  - Switch

Steps to produce microchips used in electronics



Example:



300mm wafer, 506 chips, 10nm technology
Each chip is 11.4 x 10.7 mm

How to calculate cost of Circuit?

1. **Cost per die**:

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{Yield}}$$

2. **Approximate number of dies per wafer**:

$$\text{Dies per wafer} \approx \frac{\text{Wafer area}}{\text{Die area}}$$

3. **Yield formula**:

$$\text{Yield} = \frac{1}{\left(1 + \left(\text{Defects per area} \times \frac{\text{Die area}}{2}\right)\right)^2}$$

Performance???

Airplane	Passenger capacity	Cruising range (miles)	Cruising speed (m.p.h.)	Passenger throughput (passengers × m.p.h.)
Boeing 737	240	3000	564	135,360
BAC/Sud Concorde	132	4000	1350	178,200
Boeing 777-200LR	301	9395	554	166,761
Airbus A380-800	853	8477	587	500,711

Computer performance？

- **Response time**
  - How long it takes to do a task
- **Throughput**
  - Total work done per unit time
    - e.g., tasks/transactions/… per hour
- How are response time and throughput affected by
  - Replacing the processor with a faster version?
  - Adding more processors?
- We’ll focus on response time for now…

Relative Performance

- 1. Define Performance

  Performance = 1/Execution Time

- 2. Compare the performance

  "X is n time faster than Y" means:

  P_X/P_Y = Execution Time $_Y$/ Execution Time $_X$ = n

- Example :
- We have computer A and computer B to run a program separately,

20s on A and 40s on B, how many times A is faster than B?

Execution Time$_B$ / Execution Time$_A$
= 40s / 20s = 2 → So A is 2 times faster than B

Measuring Execution Time?

1. Elapsed Time:

- **Definition**:
  Elapsed Time refers to the **total time** taken to complete a task, including all processing, waiting, and overhead time.
- **Components**:
  Elapsed Time includes:
  - **Processing Time**: The time the CPU spends actually executing the task.
  - **I/O Time**: The time spent waiting for and handling input/output operations (e.g., disk, network).
  - **OS Overhead**: Time spent on operating system tasks like scheduling and task management.
  - **Idle Time**: Time when the system is idle, waiting for resources or input.

Measuring Execution Time?

2. CPU Time:

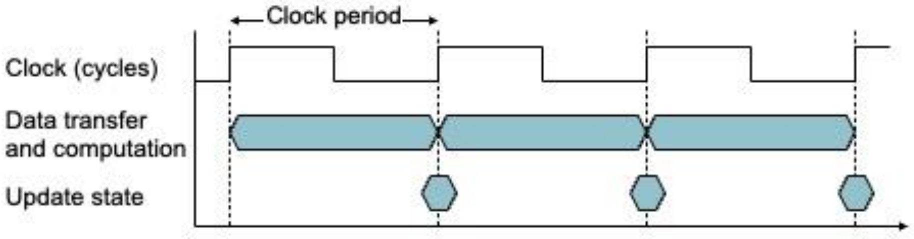
- **Definition**:
  CPU Time refers to the **pure processing time** spent by the CPU on a specific task, excluding I/O operations and time shared with other tasks.
- **Components**:
  CPU Time is composed of:
  - **User CPU Time**: The time the CPU spends executing user-level code.
  - **System CPU Time**: The time the CPU spends executing operating system code (e.g., system calls).

Summary:

- **Elapsed Time** measures the total time taken to execute a task, including CPU time, I/O time, and other system overheads. It’s useful for evaluating overall system performance.
- **CPU Time** only accounts for the time the CPU spends actively processing a task, making it a useful metric for analyzing the CPU efficiency of a program.
- Since **different programs have different dependencies on CPU and I/O performance**, optimization strategies should be tailored to the program’s nature.

CPU Clocking

- Operation of digital hardware governed by a constant-rate clock



- Clock period: duration of a clock cycle
  - e.g., 250ps = 0.25ns = 250×10^{-12}s
- Clock frequency (rate): cycles per second
  - e.g., 4.0GHz = 4000MHz = 4.0×10^{9}Hz

CPU Time

$$\text{CPU Time} = \text{CPU Clock Cycles} \times \text{Clock Cycle Time}$$

$$= \frac{\text{CPU } clock\ cycles}{Clock\ Rate}$$

Performance improved by：

Reducing number of clock cycles

Increasing clock rate

Hardware designer must often trade off clock rate against cycle count

CPU Time Example

- Computer A: 2GHz clock, 10s CPU time
- Designing Computer B
  - Aim for 6s CPU time
  - Can do faster clock, but causes 1.2 × clock cycles_A
- How fast must Computer B clock be?

CPU time example

1. Calculate the CPU clock cycles for A

$$\text{CPU time}_A = \frac{\text{CPU clock cycles}_A}{\text{Clock rate}_A}$$

$$10\,\text{seconds} = \frac{\text{CPU clock cycles}_A}{2 \times 10^9 \,\frac{\text{cycles}}{\text{second}}}$$

$$\text{CPU clock cycles}_A = 10\,\text{seconds} \times 2 \times 10^9 \,\frac{\text{cycles}}{\text{second}} = 20 \times 10^9 \text{ cycles}$$

2. Calculate the clock rate for B

$$\text{CPU time}_B = \frac{1.2 \times \text{CPU clock cycles}_A}{\text{Clock rate}_B}$$

$$6 \text{ seconds} = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{\text{Clock rate}_B}$$

$$\text{Clock rate}_B = \frac{1.2 \times 20 \times 10^9 \text{ cycles}}{6 \text{ seconds}} = \frac{0.2 \times 20 \times 10^9 \text{ cycles}}{\text{second}} = \frac{4 \times 10^9 \text{ cycles}}{\text{second}} = 4 \text{ GHz}$$

Instruction Count and CPI

- Clock Cycles = Instruction Count * Cycles per Instruction(CPI)
- CPU time = Instruction Count * CPI* Clock Cycle Time

  = Instruction Count * CPI / Clock Rate

- Instruction Count for a program
  - Determined by program, ISA and compiler
- Average cycles per instruction
  - Determined by CPU hardware
  - If different instructions have different CPI
    - Average CPI affected by instruction mix

CPI Example

- Computer A: Cycle Time = 250ps, CPI = 2.0
- Computer B: Cycle Time = 500ps, CPI = 1.2
- Same instruction amount I.
- Which is faster, and by how much?

$$\begin{aligned}
\text{CPU Time}_A &= \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A \\
&= I \times 2.0 \times 250\text{ps} = I \times 500\text{ps} \\
\text{CPU Time}_B &= \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B \\
&= I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}
\end{aligned}$$

$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{I \times 600\text{ps}}{I \times 500\text{ps}} = 1.2$$

CPI details

- If different instruction classes take different numbers of cycles
- Clock Cycles = $\sum_{i=1}^{n} CPI_i * \text{Instruction Count}_i$
- Weighted average CPI
- CPI = $\frac{Clock\ Cycles}{Insrtuction\ Count} = \sum_{i=1}^{n} CPI_i * \frac{Instruction\ Count_i}{Instrution\ Count}$

CPI Example

- Alternative compiled code sequences using instructions in classes A, B, C

- Alternative compiled code sequences using instructions in classes A, B, C

Class	A	B	C
CPI for class	1	2	3
IC in sequence 1	2	1	2
IC in sequence 2	4	1	1

- Sequence 1: IC = 5
  - Clock Cycles
    = 2×1 + 1×2 + 2×3
    = 10
  - Avg. CPI = 10/5 = 2.0
- Sequence 2: IC = 6
  - Clock Cycles
    = 4×1 + 1×2 + 1×3
    = 9
  - Avg. CPI = 9/6 = 1.5