计算机基本组成

Four Functions

• Data movement

• Data storage

• Data processing

• Control

组成

- I/O
- CPU
  - Control Unit ---- 控制单元
  - Registers ---- 寄存器
  - Arithmetic and Logic Unit(ALU)----逻辑控制单元
- Memory

冯·诺依曼计算机（Von Neumann Machine）

1. **存储结构**：指令和数据共享同一存储器。
2. **总线设计**：指令和数据共享同一总线。这意味着在任何时刻，CPU只能读取指令或数据，不能同时进行两者的访问。
3. **执行顺序**：指令通常按顺序存放和执行，但在特定条件下可以根据运算结果或设定的条件改变执行顺序。
4. **灵活性**：由于程序和数据存储在同一内存中，程序可以动态地修改或存取数据。
5. **硬件复杂性**：结构相对简单，易于实现和扩展。
6. **应用场景**：广泛应用于通用计算机、PC、服务器等。
7. **Storage Structure**: Instructions and data share the same memory.
8. **Bus Design**: Instructions and data share the same bus. This means that at any given moment, the CPU can only read instructions or data, but not both simultaneously.
9. **Execution Order**: Instructions are typically stored and executed in sequence, but under certain conditions, the execution order can be changed based on the results of computations or predefined conditions.

10. **Flexibility**: Since programs and data are stored in the same memory, programs can dynamically modify or access data.
11. **Hardware Complexity**: The structure is relatively simple, making it easy to implement and expand.
12. **Application Scenarios**: Widely used in general-purpose computers, PCs, servers, etc.

哈佛结构计算机的特点

1. **存储结构**：指令和数据存储在不同的存储器中。
2. **总线设计**：采用独立的指令总线和数据总线，允许指令和数据同时被访问。
3. **并行处理**：由于指令和数据可以并行访问，哈佛结构能够提高处理速度和效率。
4. **硬件复杂性**：需要双存储器和双总线设计，增加了硬件成本和复杂性。
5. **灵活性**：由于指令存储器通常是只读的，程序无法直接修改指令存储器中的内容。
6. **应用场景**：常用于嵌入式系统、数字信号处理器（DSP）等对性能要求较高的应用。

1. **Storage Structure**: Instructions and data are stored in separate memory units.
2. **Bus Design**: Independent instruction and data buses are used, allowing simultaneous access to instructions and data.
3. **Parallel Processing**: Since instructions and data can be accessed in parallel, the Harvard architecture can improve processing speed and efficiency.
4. **Hardware Complexity**: Requires a dual-memory and dual-bus design, which increases hardware costs and complexity.
5. **Flexibility**: As the instruction memory is typically read-only, programs cannot directly modify the contents of the instruction memory.
6. **Application Scenarios**: Commonly used in embedded systems, digital signal processors (DSPs), and other applications with high performance requirements.

总结

- **性能**：哈佛结构通过并行访问指令和数据，避免了冯·诺依曼架构中的总线冲突问题，从而提高了执行效率。
- **灵活性与成本**：冯·诺依曼架构由于其简单性和灵活性，适用于通用计算，而哈佛架构则在专用计算领域表现出色。

摩尔定律（The Era of Moore’s Law）

Double (every 18-24 months):

- The number of transistors per chip
- Computer performance : higher packing density means shorter electrical paths, giving higher performance
- CPU clock speed •Power and cooling requirements are reduced (until 2013)
- One of the above, at constant cost (until 2016)

原因

Dennard Scaling（登纳德缩放定律）是1974年由Robert Dennard提出的，它与摩尔定律共同指导了集成电路行业多年。Dennard Scaling的核心观点是，随着晶体管尺寸的缩小，其功率密度保持不变，从而使芯片的功率与芯片面积成正比

Dennard (1974) observed that voltage and current should be proportional to the linear dimensions of a transistor

- Therefore, as transistors shrank, so did required voltage and current
- power is proportional to the area of the transistor.
- 因此，随着晶体管尺寸的缩小，所需的电压和电流也随之减小.
- 功率与晶体管的面积成比例.

未来发展方向

performance / watt

operations / Joules (Joules = watts x second)