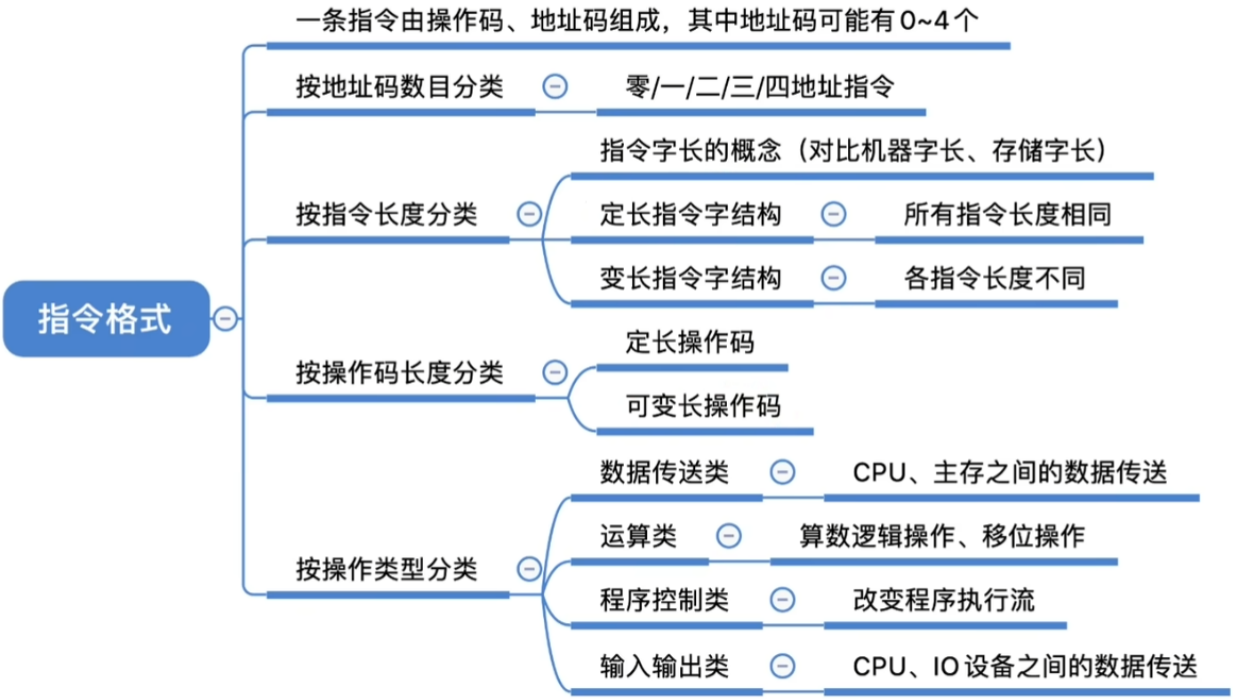
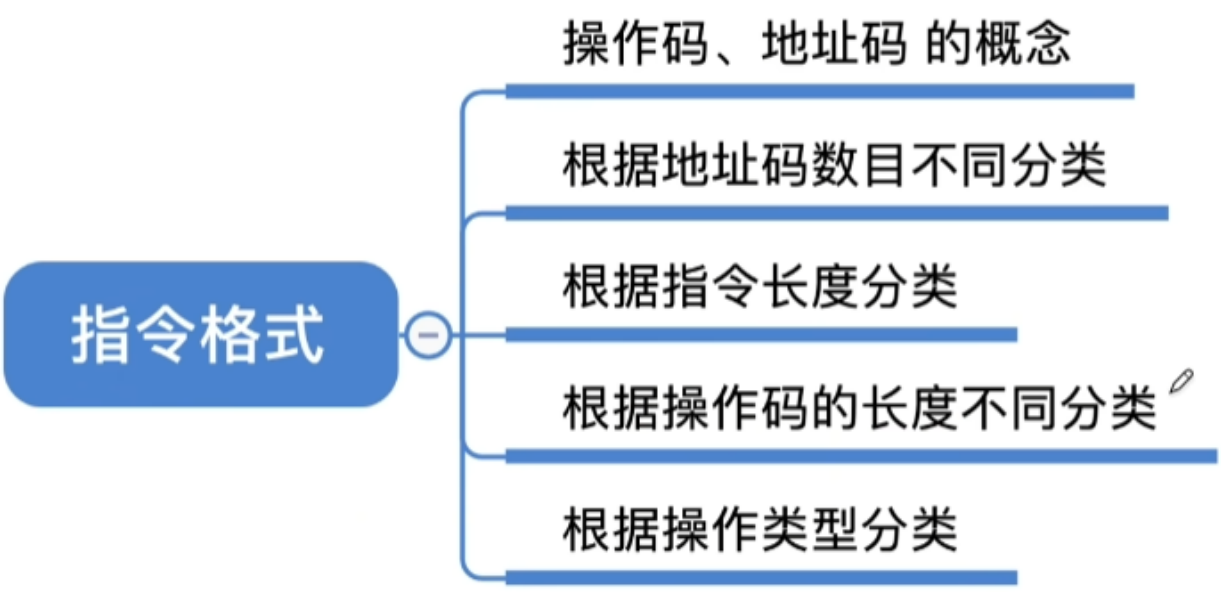
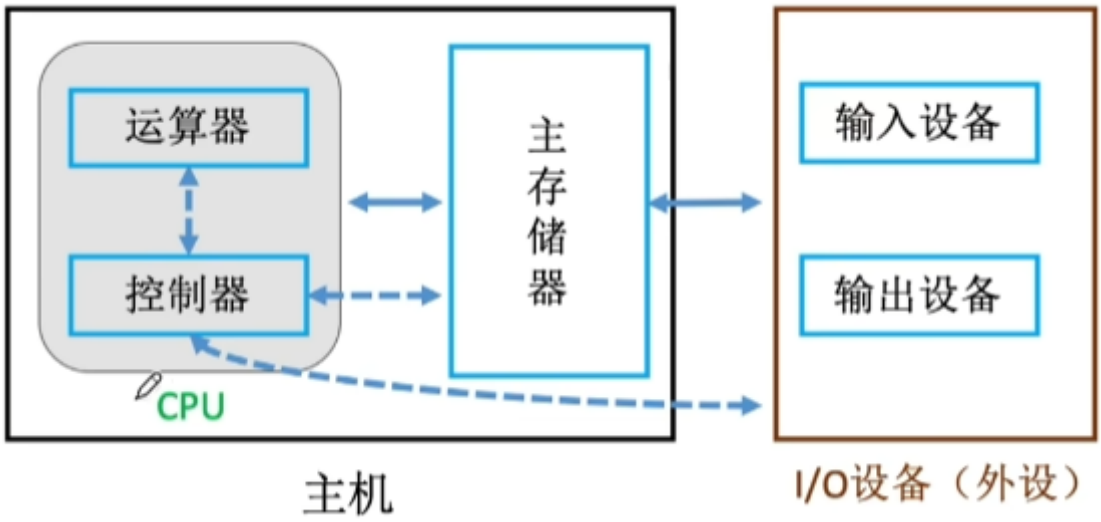
指令的基本格式





现代计算机的结构



指令的定义

指令（又称机器指令）：

是指示计算机执行某种的命令，是计算机运行的最小功能单位。

一台计算机的所有指令的集合构成该机的指令系统，也称为指令集。

注：一台计算机只能执行自己指令系统中的指令，不能执行其他系统的指令。

e.g. x86架构 ARM架构

指令格式

一条指令就是机器语言的一个语句，它是一组有意义的二进制代码。

一条指令通常要包括操作码字段和地址码字段两部分：

一条指令可能包含0个、1个、2个、3个、4个地址码

根据地址码数目不同，可以将指令分为零地址指令、一地址指令、二地址指令

按地址码数目分类

四地址指令 | OP | A_1 | A_2 | A_3（结果） | A_4（下址） |

指令含义：$(A_1)OP(A_2)\rightarrow A_3$，$A_4$=下一条将要执行指令的地址

三地址指令 | OP | A_1 | A_2 | A_3（结果） |

指令含义：$(A_1)OP(A_2)\rightarrow A_3$

二地址指令 | OP | A_1（目的操作数） | A_2（源操作数） |

指令含义：$(A_1)OP(A_2)\rightarrow A_1$

一地址指令 | OP | A_1 |

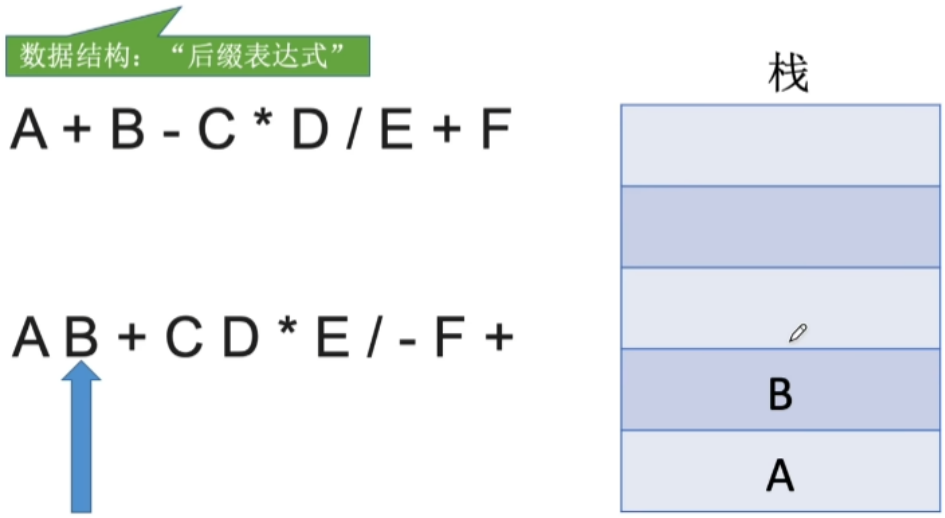
指令含义：1. $OP(A_1)\rightarrow A_1$，如加1、减1、取反、求补等
2. $(ACC)OP(A_1)\rightarrow ACC$，隐含约定的目的地址为ACC

零地址指令 | OP |

指令含义：1. 不需要操作数，如空操作、停机、关中断等指令
2. 堆栈计算机，两个操作数隐含存放在栈顶和次栈顶，计算结果压回栈顶

零地址指令

1. 不需要操作数，如空操作、停机、关中断等指令
2. 堆栈计算机，两个操作数隐含存放在栈顶和次栈顶，计算结果压回栈顶



一地址指令

1. 只需要单操作数，如加1、减、取反、求补等

$$OP(A_1) \rightarrow A_1$$

完成一条指令需要3次访存

$$\text{取指} \rightarrow \text{读}A_1 \rightarrow \text{写}A_1$$

2. 需要两个操作数，但其中一个操作数隐含在某个寄存器（如隐含在ACC）

$$(ACC)OP(A_1) \rightarrow ACC$$

完成一条指令需要2次访存

$$\text{取指} \rightarrow \text{读}A_1$$

注：A_1指某个主存地址，(A_1)表示A_1所指向的地址中的内容

二、三地址指令

常用于需要两个操作数的算术运算、逻辑运算相关指令

$$(A_1)OP(A_2) \rightarrow A_1$$

完成一条指令需要4次访存

$$\text{取指} \rightarrow \text{读}A_1 \rightarrow \text{读}A_2 \rightarrow \text{写}A_1$$

常用于需要两个操作数的算术运算、逻辑运算相关指令

$$(A_1)OP(A_2) \rightarrow A_3$$

完成一条指令需要4次访存

$$\text{取指} \rightarrow \text{读}A_1 \rightarrow \text{读}A_2 \rightarrow \text{写}A_3$$

四地址指令

$$(A_1)OP(A_2) \rightarrow A_3, A_4 = \text{下一条将要执行指令的地址}$$

完成一条指令需要4次访存

$$\text{取指} \rightarrow \text{读} A_1 \rightarrow \text{读} A_2 \rightarrow \text{写} A_3$$

正常情况下 :取指令之后$PC+1$, 指向下一条指令
四地址指令 :执行指令后, 将PC的值修改为A_4所指地址

$$n\text{位地址码的直接寻址范围} = 2^n$$

若指令总长度固定不变，则地址码数量越多，寻址能力越差

按指令长度分类

指令字长：一条指令的总长度（可能会变）

机器字长：CPU进行一次整数运算所能处理的二进制数据的位数（通常和 ALU 直接相关）

存储字长：一个存储单元中的二进制代码位数（通常和 MDR 位数相同）

半字长指令、单字长指令、双字长指令——指令字长是机器字长的多少倍

指令字长会影响取指令所需时间。如：机器字长=存储字长=16bit，则取一条双字长指令需要两次访存

定长指令字结构：指令系统中所有指令的长度都相等

变长指令字结构：指令系统中各种指令的长度不等

按操作码长度分类

定长操作码：指令系统中所有指令的操作码长度都相同

控制器的译码电路设计简单，但灵活性较低

$$n\text{位} \rightarrow 2^n\text{条指令}$$

可变长操作码：指令系统中各指令的操作码长度可变

控制器的译码电路设计复杂，但灵活性较高

$$\text{定长指令字结构} + \text{可变长操作码} \rightarrow \text{扩展操作码指令格式}$$

不同地址数的指令使用不同长度的操作码

按操作类型分类

数据传送类

进行主存与CPU之间的数据传送

数据传送

- LOAD 作用：把存储器中的数据放到寄存器中
- STORE 作用：把寄存器中的数据放到存储器中

运算类

算术逻辑操作

- 算术：加、减、乘、除、增1、减1、求补、浮点运算、十进制运算
- 逻辑：与、或、非、异或、位操作、位测试、位清楚、位求反

移位操作

- 算术移位、逻辑移位、循环移位（带进位和不带进位）

程序控制类

改变程序执行的顺序

转移操作

- 无条件转移 JMP
- 条件转移 JZ：结果位0，JO：结果溢出，JC：结果有进位
- 调用和返回 CALL 和 RETURN
- 陷阱（Trap）与陷阱指令

输入输出类（I/O）

进行CPU和I/O设备之间的数据传送

输入输出操作

- CPU寄存器与IO端口之间的数据传送（端口即IO接口中的寄存器）