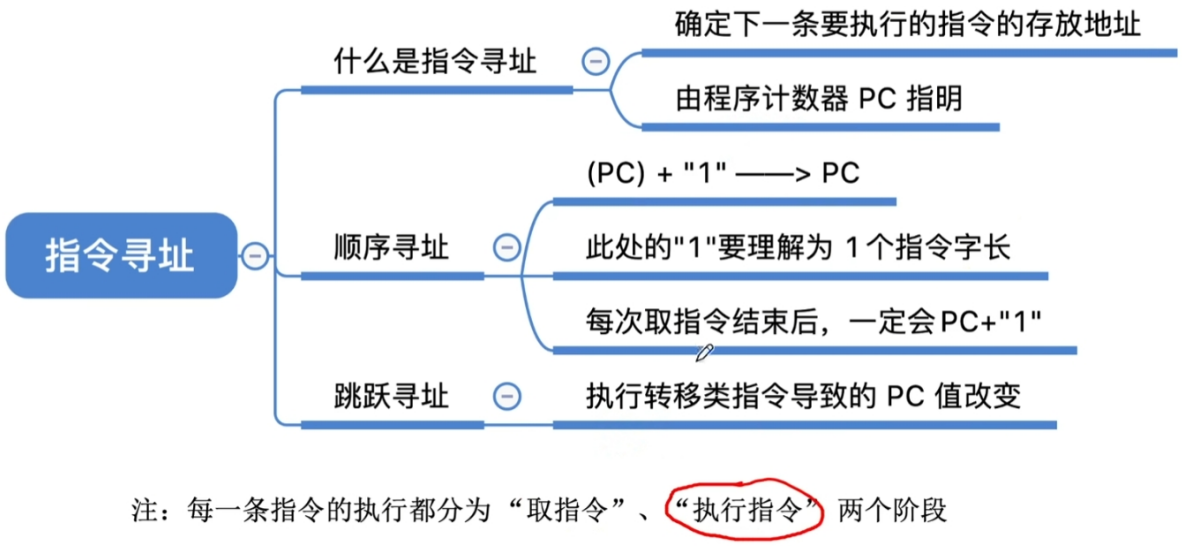
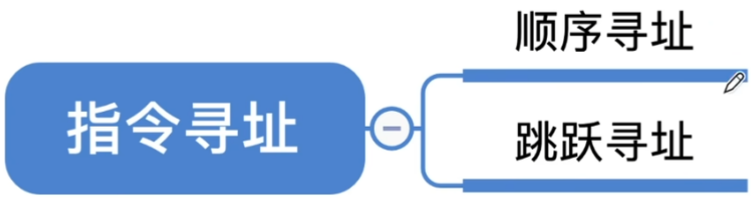
指令寻址和数据寻址

指令寻址



注：每一条指令的执行都分为“取指令”、“执行指令”两个阶段



指令寻址：如何确定下一条指令的存放地址？

一条指令的结构 | 操作码（OP） | 地址码(可能有多个)

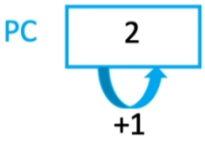
指令寻址 下一条欲执行指令的地址（始终由程序计数器PC给出）

$$下一条指令的地址：(PC)+1 \rightarrow PC$$

按字节编址怎么办?

采用变长指令字结构怎么办?

(PC) + 1 ⟶ PC



	指令地址	操作码	地址码
0		LDA	1000
1		ADD	1001
2		DEC	1200
3		JMP	7
4		LDA	2000
5		SUB	2001
6		INC	
7		LDA	1100
8		…	

该系统采用定长指令字结构

指令字长=存储字长=16bit=2B

主存按字编址

~~(PC) + 1 ⟶ PC~~

(PC) + 2 ⟶ PC

	指令地址
0	0001001111101000
2	0011001111101001
4	0010010010110000
6	1001000000000111
8	0001011111010000
10	0100011111010001
12	0101011111010001
14	0001100111000100
…	…

该系统采用定长指令字结构

指令字长=存储字长=16bit=2B

主存按字节编址

读入一个字，根据操作码判断这条指令的总字节数 n，修改PC的值

(PC) + n ⟶ PC

根据指令的类型，CPU可能还要进行多次访存，每次读入一个字

	指令地址
0	0001001111101000
2	0011001111101001
4	0010010010110000
6	1001000000000111
8	0001011111010000
10	0100011111010001
12	0101011111010001
14	0001100111000100
…	…

该系统采用变长指令字结构

指令字长=存储字长=16bit=2B

主存按字节编址

顺序寻址　(PC) + “1” ⟶ PC

跳跃寻址　由转移指令指出

	指令地址	操作码	地址码
0		LDA	1000
顺序寻址 1		ADD	1001
顺序寻址 2		DEC	1200
顺序寻址 3		JMP	7
4		LDA	2000
5		SUB	2001
6		INC	
跳跃寻址 7		LDA	1100
8		…	

该系统采用定长指令字结构

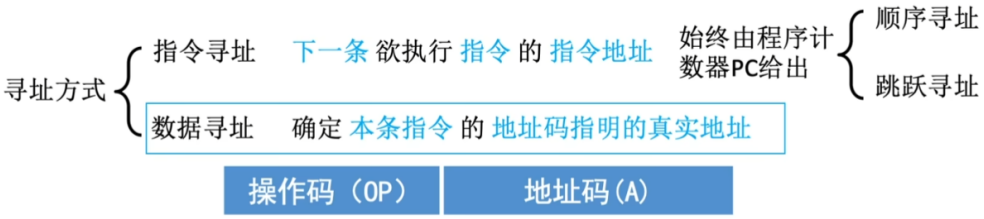
指令字长=存储字长=16bit=2B

主存按字编址

JMP：无条件转移
把PC中的内容改成7

无条件转移指令，类似C语言的 goto

指令寻址v.s.数据寻址



0	LDA	1000
1	ADD	1001
2	DEC	1200
3	JMP	7
4	LDA	2000
5	SUB	2001
6	INC	
7	LDA	1100
8	...	

100	LDA	1000
101	ADD	1001
102	DEC	1200
103	JMP	7
104	LDA	2000
105	SUB	2001
106	INC	
107	LDA	1100
108	...	

100	LDA	1000
101	ADD	1001
102	DEC	1200
103	JMP	3
104	LDA	2000
105	SUB	2001
106	INC	
107	LDA	1100
108	...	