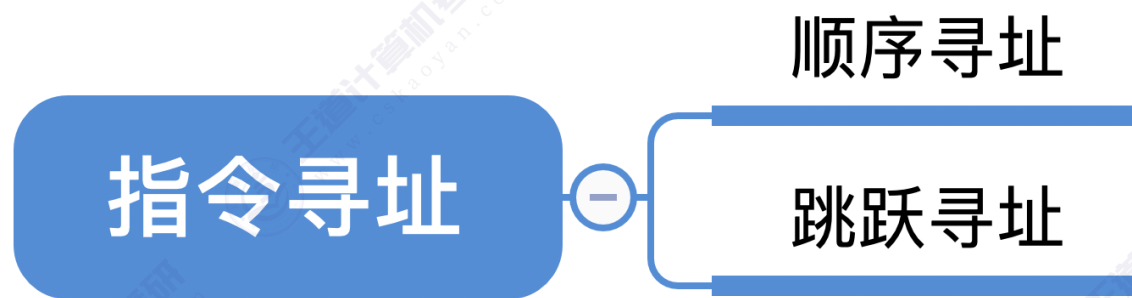


本节内容

指令寻址

知识总览



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)

回忆：计算机的工作过程

```
int a=2,b=3,c=1,y=0;
void main(){
    y=a*b+c;
}
```

程序计数器 PC:
指明下一条指令
的存放地址

PC

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?

主存 地址	指令		注释
	操作码	地址码	
0	000001	0000000101	取数 a 至ACC
1	000100	0000000110	乘 b 得 ab ,存于ACC中
2	000011	0000000111	加 c 得 $ab+c$,存于ACC中
3	000010	0000001000	将 $ab+c$,存于主存单元
4	000110	0000000000	停机
5	00000000000000010		原始数据 $a=2$
6	00000000000000011		原始数据 $b=3$
7	00000000000000001		原始数据 $c=1$
8	00000000000000000		原始数据 $y=0$

存储字长
=16bit

指令寻址

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）

$(PC) + 1 \rightarrow PC$

指令地址

0	0001001111101000
1	0011001111101001
2	0010010010110000
3	1001000000000111
4	0001011111010000
5	0100011111010001
6	0101011111010001
7	0001100111000100
8	...

该系统采用 定长指令字结构

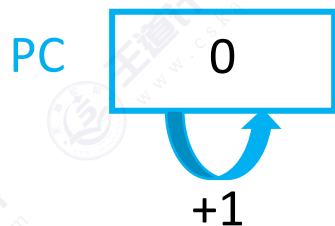
指令字长=存储字长=16bit=2B

主存 按字编址

指令寻址

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）

$(PC) + 1 \rightarrow PC$



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

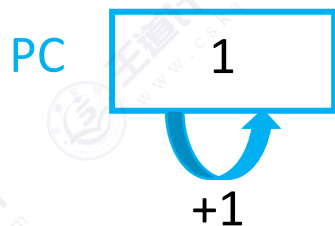
指令字长=存储字长=16bit=2B

主存按字编址

指令寻址

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）

$(PC) + 1 \rightarrow PC$



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

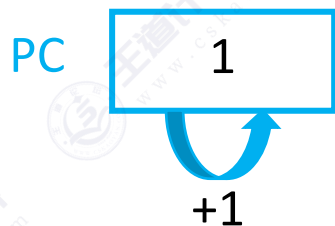
指令字长=存储字长=16bit=2B

主存按字编址

指令寻址

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）

$(PC) + 1 \rightarrow PC$



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

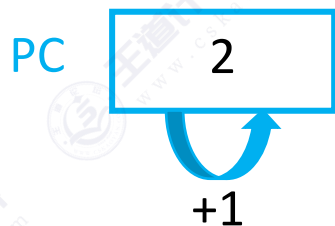
指令字长=存储字长=16bit=2B

主存按字编址

指令寻址

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）

$(PC) + 1 \rightarrow PC$



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

指令寻址

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）

~~(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

指令寻址

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）

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

$(PC) + n \rightarrow PC$

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

指令寻址



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）

顺序寻址

$(PC) + "1" \rightarrow PC$

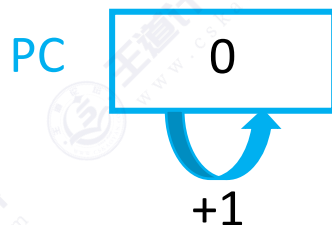
这里的1 理解为1个指令字长，实际加的值会因指令长度、编址方式而不同

指令寻址

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）

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
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

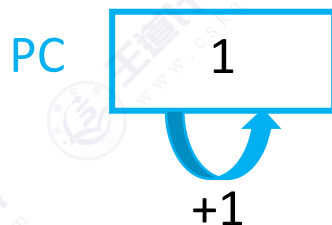
主存按字编址

指令寻址

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）

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
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

指令寻址

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)

顺序寻址

$(PC) + "1" \rightarrow PC$

跳跃寻址

由转移指令指出

下一条应该执行的指令

PC

1

+1

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	...	

当前正在执行的指令

该系统采用定长指令字结构

指令字长=存储字长=16bit=2B

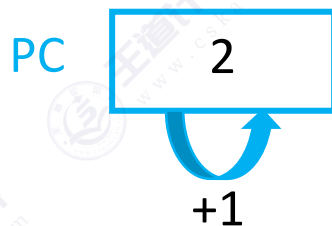
主存按字编址

指令寻址

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）

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
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

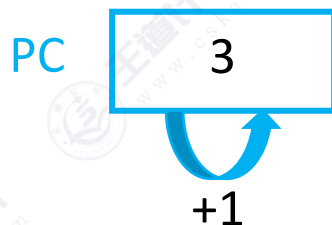
主存按字编址

指令寻址

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）

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
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

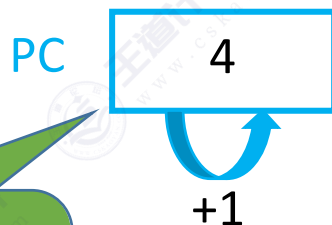
主存按字编址

指令寻址

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）

顺序寻址 $(PC) + "1" \rightarrow PC$

跳跃寻址 由转移指令指出



每次取指令之后，
PC一定会自动+1，
指向下一条应该执行的指令

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

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

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）

顺序寻址 $(PC) + "1" \rightarrow PC$

跳跃寻址 由转移指令指出

PC

7

+1

执行转移指令，将
PC值修改为7

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

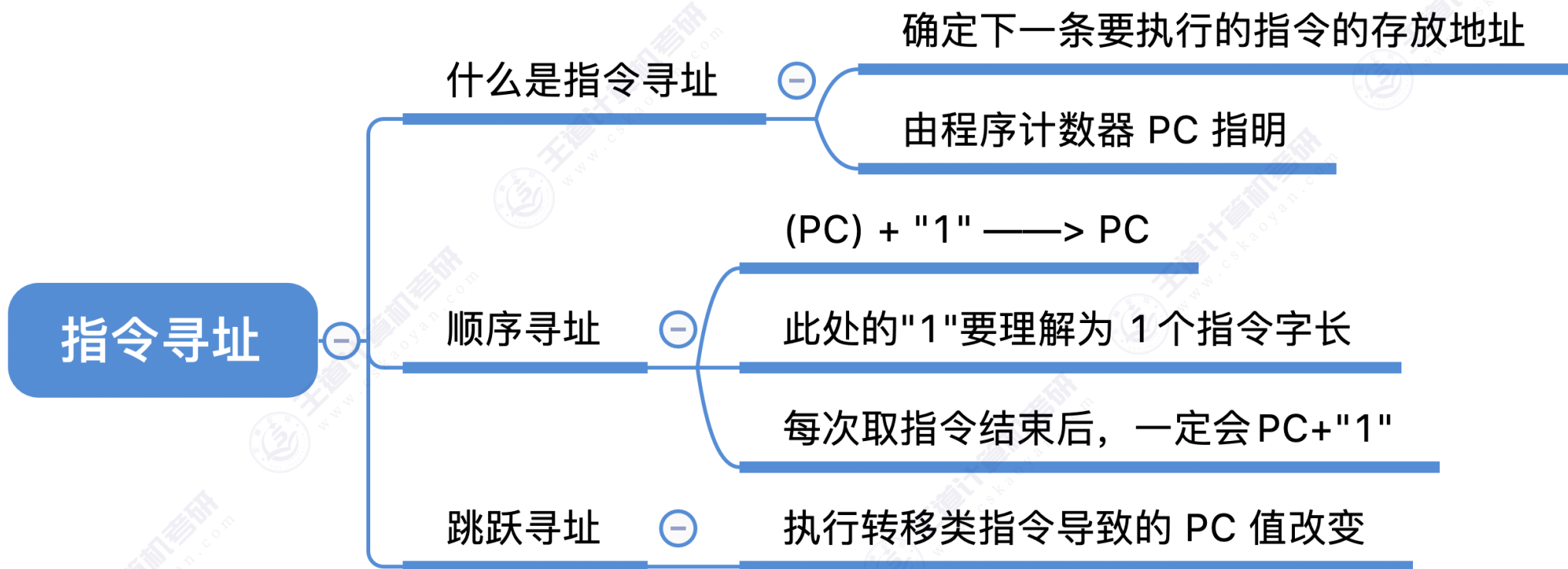
指令字长=存储字长=16bit=2B

主存按字编址

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

本节回顾



注：每一条指令的执行都分为“取指令”、“执行指令”两个阶段



公众号：王道在线



b站：王道计算机教育



抖音：王道计算机考研