



顺序寻址

指令寻址

跳跃寻址

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)

# 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:

$$(PC) + 1 \longrightarrow PC$$



按字节编址怎么办? 采用变长指令字结构怎么办?

		<u> </u>		
	主存	指令		A) EX
	地址	操作码	地址码	注释
	0	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
4 000		000110	0000000000	停机
	5	000000000000000000000000000000000000000		原始数据a=2
	6	0000000000000011		原始数据 <i>b=3</i>
	7	0000000000000000000001		原始数据 $c=1$
.46	8	0000000000000000		原始数据y=0
	3.			

存储字长 =16bit

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)

 $(PC) + 1 \longrightarrow PC$ 

指令地址

2

4

5

6

8

0 0001001111101000

1 0011001111101001

0010010010110000

3 | 100100000000111

0001011111010000

0100011111010001

0101011111010001

0001100111000100

• • •

. 2

该系统采用定长指令字结构

指令字长=存储字长=16bit=2B

5

6

8

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)

 $(PC) + 1 \longrightarrow PC$ 

PC 0 +1

指令地址 操作码 地址码

LDA	1000
ADD	1001
DEC	1200
JMP	7
LDA	2000
SUB	2001
INC	
LDA	1100
**************************************	•••

该系统采用<mark>定长指令字结构</mark>

指令字长=存储字长=16bit=2B

5

6

8

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)

 $(PC) + 1 \longrightarrow PC$ 

PC 1 +1

指令地址 操作码 地址码

LDA	1000
ADD	1001
DEC	1200
JMP	7
LDA	2000
SUB	2001
INC	
LDA	1100
** 3 4 5 ° ·	•••

该系统采用<mark>定长指令字结构</mark>

指令字长=存储字长=16bit=2B

5

6

8

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)

 $(PC) + 1 \longrightarrow PC$ 

PC 1 +1

指令地址 操作码 地址码

LDA	1000
ADD	1001
DEC	1200
JMP	7.
LDA	2000
SUB	2001
INC	
LDA	1100
1 3 ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° °	

该系统采用<mark>定长指令字结构</mark>

指令字长=存储字长=16bit=2B

5

6

8

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)

 $(PC) + 1 \longrightarrow PC$ 

PC 2 +1

指令地址 操作码 地址码

LDA	1000
ADD	1001
DEC	1200
JMP	7.
LDA	2000
SUB	2001
INC	
LDA	1100
	••

该系统采用<mark>定长指令字结构</mark>

指令字长=存储字长=16bit=2B

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)

$$(PC) + 1 \longrightarrow PC$$

$$\frac{(PC) + 1 \longrightarrow PC}{(PC) + 2 \longrightarrow PC}$$

#### 指令地址

6

8

10

12

14

 0001100111000100
0101011111010001
0100011111010001
0001011111010000
100100000000111
0010010010110000
0011001111101001
0001001111101000

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 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

$$(PC) + n \longrightarrow PC$$

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	100100000000111
8	0001011111010000
10	0100011111010001
12	0101011111010001
14	0001100111000100
	···

该系统采用<mark>变长</mark>指令字结构 <del>指令字长=存储字长=16bit=2B</del> 主存按<mark>字节</mark>编址

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)

<mark>顺序寻址</mark> (PC)+"1" → PC

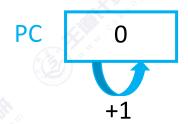
这里的1理解为1个指令字长,实际加的值会因指令长度、编址方式而不同

指令寻址 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		

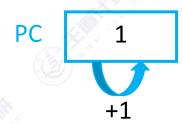
该系统采用<mark>定长指令字结构</mark> 指令字长=存储字长=16bit=2B



(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	300	••

该系统采用<mark>定长指令字结构</mark> 指令字长=存储字长=16bit=2B

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

指令地址 操作码 地址码

跳跃寻址

由转移指令指出

下一条应该执行的指令

3

5

6

8

PC 1 +1

LDA	1000
ADD	1001
DEC	1200
JMP	7
LDA	2000
SUB	2001
INC	
LDA	1100

行的指令

当前正在执

该系统采用定长指令字结构

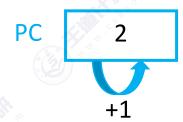
指令字长=存储字长=16bit=2B



(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		

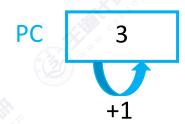
该系统采用定长指令字结构 指令字长=存储字长=16bit=2B



(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	7.7
4	LDA	2000
5	SUB	2001
6	INC	
7	LDA	1100
8	3	

该系统采用定长指令字结构 指令字长=存储字长=16bit=2B

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

PC 4 +1

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

1百令地址 探作的 地址的	指令地址	操作码	地址码
---------------	------	-----	-----

0	LDA	1000
1	ADD	1001
2	DEC	1200
3	JMP	J/07
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 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  $(PC) + "1" \longrightarrow PC$ 

由转移指令指出 跳跃寻址

PC

执行转移指令,将 PC值修改为7

指令地址 操作码 地址码

顺序寻址 1 顺序寻址 2

顺序寻址 3

5

6

跳跃寻址 7

8

LDA	1000	
ADD	1001	
DEC	1200	
JMP	7	
LDA	2000	
SUB	2001	
INC		
LDA	1100	

该系统采用定长指令字结构

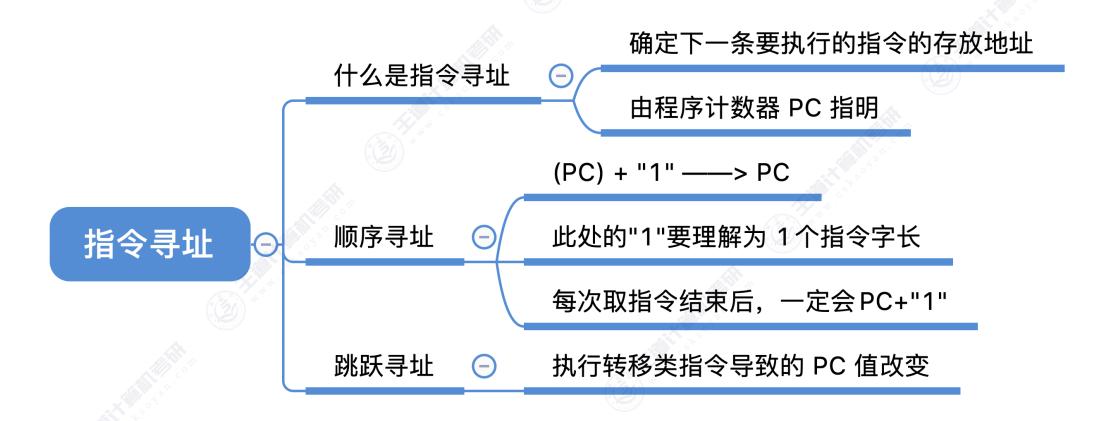
指令字长=存储字长=16bit=2B

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