

# 硬布线控制器

所有指令的取指周期、 $T_0$ 节拍下一定  
要完成 $(PC) \rightarrow MAR$ 。则可知  $C_1 = FE \cdot T_0$



Tips: 逻辑表达式是  
电路的数学化描述

## 硬布线控制器

每个时钟周期发出一个节拍信号  
(循环发出)

每个时钟周期发出一个节拍信号  
(循环发出)

指令寄存器 IR  
n位操作码

操作码译码器

$\downarrow$

$\downarrow$

$T_0 \downarrow 1 \cdots \downarrow 2^{n-1}$

FE、IND、EX、INT (事实上这  
四个触发器集成在CU内部)

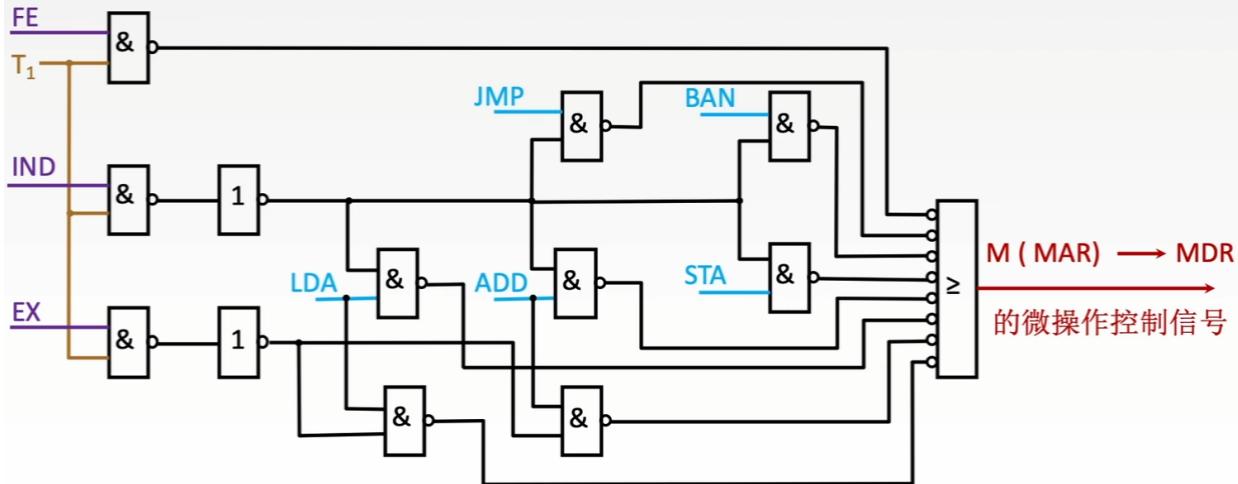
标志(来自执行单元的反馈信息)

可能来自运算器的  
PSW、ACC的符号  
位等; 也可能来自  
I/O设备、主存

如: 要让  $C_1$  对应微操作  $(PC) \rightarrow MAR$ ,  
则将其接到  $PC_{out}$ 、 $MAR_{in}$  即可

根据 指令操作码、目前的机器周期、节拍信号、机器状态条件, 即可确定  
现在这个节拍下应该发出哪些“微命令”

**M (MAR) → MDR 微操作命令的逻辑表达式:**  
 $FE \cdot T_1 + IND \cdot T_1(ADD+STA+LDA+JMP+BAN) + EX \cdot T_1(ADD+LDA)$



根据 指令操作码、目前的机器周期、节拍信号、机器状态条件, 即可确定  
现在这个节拍下应该发出哪些“微命令”

## 硬布线控制器的设计

### 设计步骤:

1. 分析每个阶段的微操作序列
2. 选择CPU的控制方式
3. 安排微操作时序
4. 电路设计
  - (1)列出操作时间表
  - (2)写出微操作命令的最简表达式
  - (3)画出逻辑图

硬布线控制器的特点:

指令越多, 设计和实现就越复杂, 因此一般用于 RISC (精简指令集系统)

如果扩充一条新的指令, 则控制器的设计就需要大改, 因此扩充指令较困难。

由于使用纯硬件实现控制, 因此执行速度很快。微操作控制信号由组合逻辑电路即时产生。

设计步骤:

确定哪些指令在什么阶段、在什么条件下会使用到的微操作

1. 分析每个阶段的微操作序列 (取值、间址、执行、中断四个阶段)

2. 选择CPU的控制方式

采用定长机器周期还是不定长机器周期? 每个机器周期安排几个节拍?

3. 安排微操作时序

如何用3个节拍完成整个机器周期内的所有微操作?

4. 电路设计

确定每个微操作命令的逻辑表达式, 并用电路实现

假设采用同步控制方式 (定长机器周期), 一个机器周期内安排3个节拍。

安排, 必须安排



## 分析每个阶段的微操作序列

注: 中断周期内的微操作序列就不分析了, 原理类似

### 分析每个阶段的微操作序列

取指周期 (所有指令都一样)

PC → MAR

1 → R

M ( MAR ) → MDR

MDR → IR

OP ( IR ) → ID

( PC ) + 1 → PC

间址周期 (所有指令都一样)

Ad( IR ) → MAR

1 → R

M ( MAR ) → MDR

MDR → Ad( IR )

注: ID 是指令译码器  
Instruction Decoder

执行周期 (各不相同)

CLA

clear ACC 指令

ACC 清零

LDA X

取数指令,  
把X所指内容  
取到ACC

注: 很多地方把  
ACC 简写为 AC

0 → AC

1 → R

M ( MAR ) → MDR

MDR → AC

罗列出所有指令在各个阶段的微操作序列, 就可以知道在什么情况下需要使用这个微操作

根据 指令操作码、目前的机器周期、节拍信号、机器状态条件, 即可确定现在这个节拍下应该发出哪些“微命令”

JMP X  
无条件转移

Ad ( IR ) → PC

负数符号位为1

BAN X

Ad ( IR ) + A0 • ( PC ) → PC

Branch ACC Negative

条件转移, 当ACC为负时转移

## 安排微操作时序

### 安排微操作时序的原则

原则一 微操作的 先后顺序不得 随意 更改

原则二 被控对象不同 的微操作

尽量安排在 一个节拍 内完成

原则三 占用 时间较短 的微操作

尽量 安排在 一个节拍 内完成

并 允许有先后顺序

### 取指周期

(1) PC → MAR

(2) 1 → R 存储器空闲即可

(3) M ( MAR ) → MDR 在(1)之后

(4) MDR → IR 在(3)之后

(5) OP ( IR ) → ID 在(4)之后

(6) <sup>0</sup> ( PC ) + 1 → PC 在(1)之后

原则一 微操作的 <b>先后顺序不得</b> 随意 <b>更改</b>	$T_0$ (1) $PC \rightarrow MAR$	
原则二 <b>被控对象不同</b> 的微操作 尽量安排在 <b>一个节拍</b> 内完成	$T_0$ (2) $1 \rightarrow R$	存储器空闲即可
原则三 占用 <b>时间较短</b> 的微操作 尽量 安排在 <b>一个节拍</b> 内完成 并允许有先后顺序	$T_1$ (3) $M(MAR) \rightarrow MDR$ $T_1$ (6) $(PC) + 1 \rightarrow PC$ $T_2$ (4) $MDR \rightarrow IR$ $T_2$ (5) $OP(IR) \rightarrow ID$	在(1)之后 在(1)之后 在(3)之后 在(4)之后

两个微操作占用时  
间较短，根据原则  
三安排在一个节拍

**M ( MAR ) → MDR** 从主存取数据，用时较长，因此必须一个时钟周期才能保证微操作的完成

**MDR → IR** 是CPU内部寄存器的数据传送，速度很快，因此在一个时钟周期内可以紧接着完成 **OP ( IR ) → ID**。  
也就是可以一次同时发出两个微命令。

## 间址周期

$T_0$	(1) $Ad(IR) \rightarrow MAR$
$T_0$	(2) $1 \rightarrow R$
$T_1$	(3) $M(MAR) \rightarrow MDR$
$T_2$	(4) $MDR \rightarrow Ad(IR)$

## 执行周期

## 安排微操作时序-执行周期

### (1) 非访存指令

① CLA	$T_0$					
clear	$T_1$					
ACC清零	$T_2$	$0 \rightarrow AC$				
② COM	$T_0$					
complement	$T_1$					
ACC取反	$T_2$	$\overline{AC} \rightarrow AC$				
③ SHR	$T_0$					
shift	$T_1$					
算术右移	$T_2$	$L(AC) \rightarrow R(AC)$				
		$AC_0 \rightarrow AC_0$				
④ CSL	$T_0$					
cyclic shift	$T_1$					
循环左移	$T_2$	$R(AC) \rightarrow L(AC), AC_0 \rightarrow AC_n$				
⑤ STP	$T_0$					
stop	$T_1$					
停机	$T_2$	$0 \rightarrow G$				

### (2) 访存指令

⑥ ADD X	$T_0$	$Ad(IR) \rightarrow MAR, 1 \rightarrow R$				
加法指令	$T_1$	$M(MAR) \rightarrow MDR$				
隐含ACC	$T_2$	$(AC) + (MDR) \rightarrow AC$				
⑦ STA X	$T_0$	$Ad(IR) \rightarrow MAR, 1 \rightarrow W$				
存数指令	$T_1$	$AC \rightarrow MDR$				
隐含ACC	$T_2$	$MDR \rightarrow M(MAR)$				
⑧ LDA X	$T_0$	$Ad(IR) \rightarrow MAR, 1 \rightarrow R$				
取数指令	$T_1$	$M(MAR) \rightarrow MDR$				
隐含ACC	$T_2$	$MDR \rightarrow AC$				
⑨ JMP X	$T_0$					
jump	$T_1$					
无条件转移	$T_2$	$Ad(IR) \rightarrow PC$				
⑩ BAN X	$T_0$					
Branch ACC	$T_1$					
Negative	$T_2$	$A_0 \cdot Ad(IR) + \overline{A}_0 \cdot (PC) \rightarrow PC$				
条件转移						

## 组合逻辑设计

设计步骤：

1. 列出操作时间表

列出在取指、间址、执行、中断周期， $T_0$ 、 $T_1$ 、 $T_2$  节拍内有  
可能用到的所有微操作

2. 写出微操作命令的最简表达式

3. 画出逻辑图

## 列出操作时间表

设计步骤:  
1. 列出操作时间表

非访存指令

工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	SHR	CSL	STP	ADD	STA	LDA	JMP	BAN
				1	1	1	1	1					
FE 取指	T <sub>0</sub>		PC→MAR	1	1	1	1	1	1	1	1	1	1
			1→R	1	1	1	1	1	1	1	1	1	1
	T <sub>1</sub>		M(MAR)→MDR	1	1	1	1	1	1	1	1	1	1
			(PC)+1→PC	1	1	1	1	1	1	1	1	1	1
	T <sub>2</sub>		MDR→IR	1	1	1	1	1	1	1	1	1	1
			OP(IR)→ID	1	1	1	1	1	1	1	1	1	1
	I		1→IND						1	1	1	1	1
	$\overline{I}$		1→EX	1	1	1	1	1	1	1	1	1	1

间址特征

工道学苑 www.gongdaoxueyuan.com

设计步骤:  
1. 列出操作时间表

非访存指令

工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	SHR	CSL	STP	ADD	STA	LDA	JMP	BAN
				1	1	1	1	1					
IND 间址	T <sub>0</sub>		Ad(IR)→MAR						1	1	1	1	1
			1→R						1	1	1	1	1
	T <sub>1</sub>		M(MAR)→MDR						1	1	1	1	1
			MDR→Ad(IR)						1	1	1	1	1
	$\overline{IND}$		1→EX						1	1	1	1	1

间址周期标志

组合逻辑设计

1. 列出操作时间表



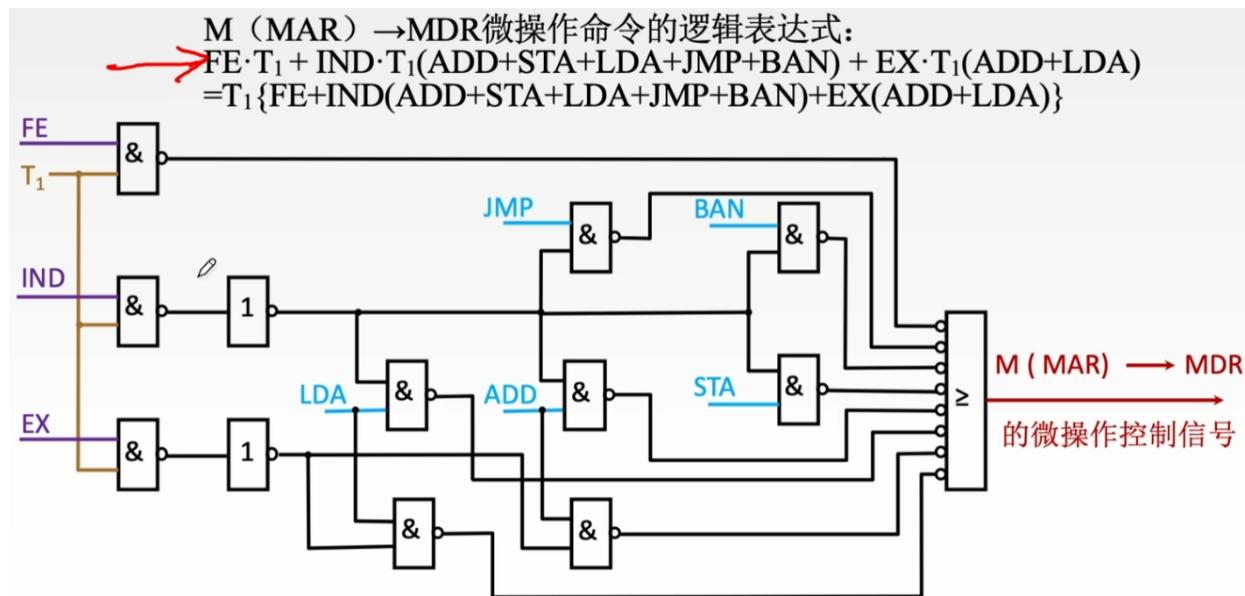
工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	ADD	STA	LDA	JMP	BAN
				1	1	1	1	1	1	1
EX 执行	T <sub>0</sub>		Ad(IR)→MAR			1	1	1		
			1→R			1		1		
			1→W				1			
	T <sub>1</sub>		M(MAR)→MDR			1		1		
			AC→MDR				1			
	T <sub>2</sub>		(AC)+(MDR)→AC			1				
			MDR→M(MAR)				1			
			MDR→AC					1		
			0→AC	1						
			$\overline{AC} \rightarrow AC$		1					
			Ad(IR)→PC						1	
			A <sub>0</sub>	Ad(IR)→PC						1

## 微操作信号综合

工作周期标记	节拍	状态条件	微操作命令信号	CLA	COM	SHR	CSL	STP	ADD	STA	LDA	JMP	BAN
FE 取指	T <sub>0</sub>		PC → MAR	1	1	1	1	1	1	1	1	1	1
			1 → R	1	1	1	1	1	1	1	1	1	1
	T <sub>1</sub>		M(MAR) → MDR	1	1	1	1	1	1	1	1	1	1
IND 间址	T <sub>1</sub>									1	1	1	1
EX 执行	T <sub>1</sub>									1	1		

$M(MAR) \rightarrow MDR$  微操作命令的逻辑表达式:  
 $FE \cdot T_1 + IND \cdot T_1(ADD+STA+LDA+JMP+BAN) + EX \cdot T_1(ADD+LDA)$   
 $= T_1\{FE+IND(ADD+STA+LDA+JMP+BAN)+EX(ADD+LDA)\}$

## 画出逻辑图



根据 指令操作码、目前的机器周期、节拍信号、机器状态条件，即可确定  
现在这个节拍下应该发出哪些“微命令”