

第5章 控制器逻辑

5.1 知识要点

5.1.1 处理器的外特性——指令系统

指令系统是 CPU 功能的表征和软件开发的基础,也是 CPU 设计的基本依据。

1. 符号语言与汇编语言的概念

1) 数据类型

Intel 8086 汇编语言中允许使用如下形式的数值数据:

- 二进制数据,后缀为 B,如 10101011B。
- 十进制数据,后缀为 D,如 235D。
- 八进制数据,后缀为 Q(本应是 O,为避免与数字 0 相混,用 Q 代),如 235Q。
- 十六进制数据,后缀为 H,如 BAC3H。

2) 运算符

- 算术运算符: +、-、*、/。
- 关系运算符: EQ(相等)、NE(不相等)、LT(小于)、GT(大于)、LE(小于等于)、GE(大于等于)。
- 逻辑运算符: AND(“与”)、OR(“或”)、NOT(“非”)。

3) 汇编语言指令的一般形式

标号: 操作码 地址码(操作数);注释

2. 寻址方式

- (1) 立即寻址(immediate addressing)。
- (2) 寄存器直接寻址(register addressing)。
- (3) 存储器直接寻址。
- (4) 存储器间接寻址。
- (5) 变址/基址寻址。
- (6) 堆栈寻址。
- (7) 8086 的段寻址。

3. CPU 中的可编程寄存器

- (1) 数据寄存器。
- (2) 指针及地址寄存器。

(3) 段寄存器。

(4) 控制寄存器。

- IP (或 PC)——指令指针寄存器(或程序计数器)。
- EFLAGS——状态和标志寄存器,用每一位指示处理器的一种工作状态,以控制处理器的工作。

在 CPU 设计时,要根据指令系统的要求提供充分的寄存器组(寄存器堆)。

4. Intel 8086 指令分类

(1) 数据传送类指令。

- 通用数据传送指令。
- 输入输出指令。
- 地址传送指令。
- 标志传送指令。

(2) 算术运算指令。

(3) 逻辑运算指令。

(4) 移位指令和循环移位指令。

(5) 串操作指令。

(6) 程序控制指令。

- JMP: 无条件转移。
- CALL: 过程调用指令。
- RET: 过程返回指令。

5. 指令系统的设计内容

- 操作类型: 决定指令数的多少和操作的难易程度。
- 数据类型: 决定操作数的存取方式。
- 寻址方式: 寻找操作数和转移地址的方式。
- 寄存器类型及个数。
- 指令格式: 包括指令长度和各字段的布局等。

6. CISC 与 RISC

重点掌握以下知识点:

- (1) 冯·诺依曼语义差距问题。
- (2) 80-20 规律。
- (3) RISC 的基本思想。

5.1.2 指令的时序

1. 指令时序的控制方式

- (1) 同步控制方式: “以时定序”。

(2) 异步控制方式：“以序定时”。

2. 指令的微操作分析

不同的指令含有不同的微操作,形成不同的指令周期结构。

3. 指令周期的控制

(1) 集中控制：“统一划齐”的控制方式。

(2) 分散的节拍控制：“按需分配”控制的方式。

(3) 集中/局部混合控制：折中方案,考虑绝大多数指令的需要。

5.1.3 控制器设计

1. 控制器的基本组成

(1) CPU 寄存器组。

(2) 操作控制部件。

(3) 时序部件,即节拍发生器。

2. 组合逻辑控制器

采用开关理论中时序电路的方法设计微操作控制线路。步骤如下：

(1) 分解每条指令,归纳成若干基本操作。

(2) 将各微操作落实到指令周期的不同节拍中去,即编排操作时间表。

(3) 对全部指令的操作时间表进行综合、分析,求出各相同的微操作产生条件的逻辑表达式,并画出操作控制线路。

3. 微程序控制器

1) 微程序控制的基本思想

把一条指令看作是由一个微指令系列组成的微程序。

由于每一条指令所包含的微指令序列是固定的,通常把指令系统存储在专门的 ROM 中,并称之为控制存储器。显然,执行一条指令的关键,是找到它在控制存储器中的入口地址。

2) 微程序控制器的组成

- 控制存储器。
- 微指令寄存器和微地址形成电路。
- 指令操作码译码器。

5.2 习题解析

5.1 某机器字长 32 位,指令单字长,地址字段长 12 位,指令系统中没有三地址指令,采用变长操作码。求该计算机中的二地址指令、一地址指令和零地址指令最多各有多少条。

解：根据题意，分析如下：

(1) 对于二地址指令，操作码长度为 $32 - 2 \times 12 = 8(\text{b})$ ，即编码空间为 2^8 。由于要使用一个码作为扩展窗口，所以该计算机中的二地址指令最多有 $2^8 - 1$ 条。

(2) 一地址指令最多的情况是指令系统中只有一条二地址指令，并要为零地址指令留下一个扩展窗口。首先可以得到在 OP 字段中给一地址指令留下的扩展窗口为 $(2^8 - 1)$ 个。故一地址指令编码空间为 $(2^8 - 1) \times 2^{12}$ 。但还要为零地址指令留下一个扩展窗口，所以最多可以有 $(2^8 - 1) \times 2^{12} - 1 = 2^{20} - 2^{12} - 1$ 条一地址指令。这个结果表明，一条二地址指令需要“牺牲” 2^{12} 条一地址指令。或者说，少一条二地址指令，就可以增加 2^{12} 条一地址指令。

(3) 零地址指令最多的情况是指令系统中只有一条二地址指令和一条一地址指令，其余均为零地址指令。由(2)可知，在只有一条二地址指令的情况下，一地址指令的编码空间为 $(2^8 - 1) \times 2^{12}$ 。在只有一条一地址指令的情况下，为零地址指令留下的扩展窗口为 $(2^8 - 1) \times 2^{12} - 1$ 。而零地址指令的操作码又在一地址指令留下的扩展窗口基础上扩展了 12b，所以零地址指令的编码空间为 $((2^8 - 1) \times 2^{12} - 1) \times 2^{12} = 2^{32} - 2^{24} - 12^{12}$ 。

5.2 某计算机字长 16 位，指令中的地址码长 6 位，有零地址指令、一指令地址和二地址指令 3 种指令。若二地址指令有 K 条，零地址指令有 L 条，则一地址指令最多可以有多少条？

解：设一地址指令最多可以有 X 条，则：

- 二地址指令给一地址指令留下的扩展窗口为 $2^4 - K$ 。
- 一地址指令给零地址指令留下的扩展窗口为 $(2^4 - K) \times 2^6 - X$ 。
- 零地址指令的条数为 $L = ((2^4 - K) \times 2^6 - X) \times 2^6$ 。

故， $X = 2^{10} - K \times 2^6 - L/2^6$ 。

$L/2^6$ 带小数时向正 ∞ 舍入。

5.3 指令系统中采用不同寻址方式的主要目的是_____。

- A. 实现程序控制和快速查找存储器地址
- B. 可以直接访问主存和外存
- C. 缩短指令长度，扩大寻址空间，提高编程的灵活性
- D. 降低指令译码难度

解：选 C。

对 A，程序控制是由控制器中的地址指示器来完成的，增加了寻址方式会降低查找存储器地址的速度。

对 B，无论有无多种寻址方式，CPU 均能直接访问主存，绝对不能直接访问外存。

对 D，指令译码难度与操作数的个数、寻址方式及指令的长度有关，但寻址方式越多，译码就越复杂，因而不会降低译码的难度。

5.4 某计算机指令格式如下：

θ	λ	D
----------	-----------	-----

其中， θ 为操作码，代表如下一些操作：

- λ 为寻址方式,代表如下一些寻址方式:

- D 为形式地址,变址寄存器内容为 0005H。

今有如下程序：

请把上述程序执行后存储器各单元的内容填入表 5.1 中。

表 5.1 程序执行后存储器各单元的内容

[illegible]

解：执行第 1 条指令，寻址方式为变址间址方式，操作数地址 = (变址寄存器 + 偏移地址) = (0005H + 0005H) = (0AH) = 08H。把 08H 单元的内容取到累加器 A 中，即 A 存放了 06H。

执行第 2 条指令,寻址方式为间址变址方式,操作数地址=(偏移地址)+(变址寄存器)
=(0006H)+0005H=0004H+0005H=0009H,操作数则为 09H 单元的内容。该指令把累加器 A 的内容与 09H 单元的内容相加得到 0DH,并把结果存入累加器 A 中。

执行第 3 条指令,寻址方式为立即寻址,即指令中给出的就是操作数,该指令把累加器 A 的内容加上数 0007H,得到 14H,并把结果存入累加器 A 中。

执行第 4 条指令,寻址方式为间接寻址,指令中给出的偏移量是操作数地址的地址,即:
操作数地址=(0008H)=06H,该指令把累加器的内容传送到 06H 单元中,(06H)=14H。

执行第 5 条指令,寻址方式为直接寻址,指令中给出的是操作数的地址 0007H,该指令把累加器 A 的内容与 007H 单元的内容相加,得到 19H,并把结果存入累加器 A 中。

执行第 6 条指令,寻址方式为变址寻址,操作数地址=(变址寄存器)+偏移量=0005H+0006H=000BH,该指令把累加器 A 的内容传送到 000BH 单元中,(000BH)=19H。

上述指令中只有两条指令涉及存储器写操作,也就是只有两个单元的内容有变化,最后结果见表 5.2。

表 5.2 程序执行后存储器各单元的内容

地 址	存储器单元内容		
	程序执行前	程序执行后	
	十六进制	十六进制	十进制
0004H	02H	02H	02
0005H	03H	03H	03
0006H	04H	04H	04
0007H	05H	05H	05
0008H	06H	06H	06
0009H	07H	07H	07
000AH	08H	08H	08
000BH	09H	09H	09
000CH	0AH	0AH	10
000DH	0BH	0BH	11
000EH	0CH	0CH	12
000FH	0DH	0DH	13

5.5 在 8086 中,对于物理地址 2014CH 来说,如果段起始地址为 20000H,则偏移量应为多少?

解: 在 8086 中,物理地址=段起始地址+偏移量,因此偏移量=物理地址-段起始地址=2014H-20000H=014CH。

5.6 在 8086 中,SP 的初值为 2000H,AX=3000H,BX=000H。

(1) 计算执行指令 PUSH AX 后 SP 的值。

(2) 计算再执行指令 PUSH BX 及 POP AX 后 SP 和 BX 的值,请画出堆栈变化示意图。

解:

(1) 在 8086 中,堆栈的指针是向下增大的,即栈底地址是高地址,栈顶地址为低地址。因此在数据压入堆栈后,栈顶指针减小,本指令压入的是 16 位二进制数, $SP = 2000 - 2 = 1FFE$ 。

(2) 执行 PUSH BX 及 POP AX 后,栈顶指针先 -2 后 $+2$, $SP = 1FFE$,BX 寄存器内容无变化, $BX = 5000$ 。堆栈变化示意图如图 5.1 所示。

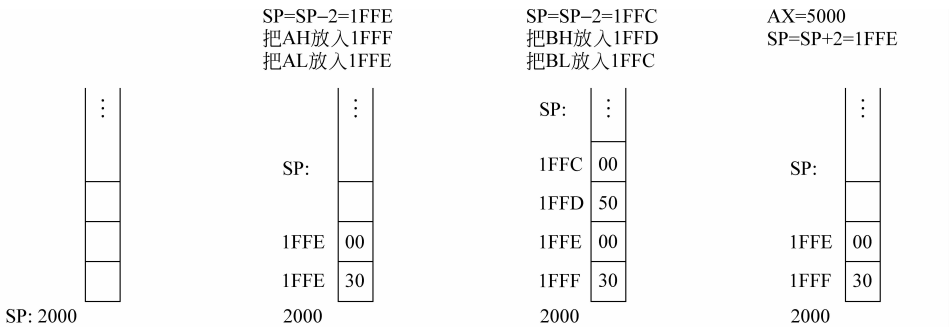


图 5.1 堆栈变化

5.7 指出下列 8086 指令中源操作数和目的操作数的寻址方式。

- (1) PUSH AX
- (2) XCHG BX, [BP+SI]
- (3) MOV CX, 03F5H
- (4) LDS SI, [BX]
- (5) LEA BX, [BX+SI]
- (6) MOV AX, [BX+SI+0123H]
- (7) MOV CX, ES:[BX][SI]
- (8) MOV [SI], AX

解:

- (1) 操作数为寄存器寻址方式。
- (2) 源操作数为基址变址寻址方式,目的操作数为寄存器寻址方式。
- (3) 源操作数为立即寻址方式,目的操作数为寄存器寻址方式。
- (4) 源操作数为寄存器间接寻址方式,目的操作数为寄存器寻址方式。
- (5) 源操作数为基址变址寻址方式,目的操作数为寄存器寻址方式。
- (6) 源操作数为基址变址寻址方式,目的操作数为寄存器寻址方式。
- (7) 源操作数为基址变址寻址方式,目的操作数为寄存器寻址方式。
- (8) 源操作数为寄存器间接寻址方式,目的操作数为基址变址寻址方式。

5.8 指出下列 8086 指令中有关转移地址的寻址方式。

- (1) JMP WORD PTR[BX][SI]
- (2) JMP SHORT SUB1
- (3) JMP DWORD PTR [BX+SI]

解:

(1) 段内间接转移,寻址方式为基址变址寻址方式。PTR 是属性操作码。

(2) 段内直接短转移,寻址方式为存储器直接寻址方式。

(3) 段间间接转移,寻址方式为基址变址寻址方式。

5.9 在 8086 中,给定(BX)=362AH,(SI)=7B9CH,偏移量 D=3C25H,试确定在以下各种寻址方式下的有效地址是什么。

(1) 立即寻址。

(2) 直接寻址。

(3) 使用 BX 的寄存器寻址。

(4) 使用 BX 的间接寻址。

(5) 基址变址寻址。

(6) 相对基址变址寻址。

解:

(1) 在立即寻址方式中,指令中给出的 D 就是操作数。也可以说本条指令的地址就是操作数地址。

(2) 在直接寻址方式中,指令中给出的 D 是操作数的有效地址,EA=3C25H。

(3) 使用 BX 的寄存器直接寻址,有效地址就是指令中给出的寄存器号,即 BX 中存放的是操作数。EA=BX。

(4) 使用 BX 的寄存器间接寻址,有效地址是寄存器中存放的数据。EA=(BX)=362AH。

(5) 用基址变址寻址,有效地址 EA=(BX)+(SI)=362AH+7B9CH=B1C6H。

(6) 相对基址寻址方式,有效地址 EA=(BX)+(SI)+D=362AH+7B9CH+3C25H=EDEBH。

5.10 有一个主频为 25MHz 的微处理器,平均每条指令的执行时间为两个机器周期,每个机器周期由两个时钟脉冲组成。

(1) 假定存储器为零等待,请计算机器速度(每秒执行的机器指令条数)。

(2) 假如存储器速度较慢,每两个机器周期中有一个访问存储器周期,需插入两个时钟的等待时间,请计算机器速度。

解:

(1) 存储器为零等待是假设在访问存储器时存储周期等于机器周期,此时,机器周期=主振周期×2(一个机器周期由两个时钟脉冲组成):

$$1 \div (25 \times 10^6) \times 2 \times 10^6 = 0.08(\mu s)$$

指令周期为 $2 \times \text{机器周期} = 0.16\mu s$ 。

机器平均速度为 $1/0.16 \approx 6.25(\text{MIPS})$ 。

(2) 若每两个机器周期有一个是访存周期,则需要插入两个时钟的等待时间。

指令周期为 $0.16 + 0.08 = 0.24(\mu s)$ 。

机器的平均速度为 $1/0.24 \approx 4.2(\text{MIPS})$ 。

5.11 控制器有哪几种控制方式? 各有什么特点?

解：控制器有 3 种控制方式：同步控制、异步控制和联合控制。

同步控制方式是指任何指令的运行或指令中每个微操作的执行,都由确定的具有基准时钟的时序信号所控制。时序信号的结束就意味着安排的工作已经结束。同步控制要求知道每一个微操作序列的长度和微操作的时间,控制线路集中并且简单。异步控制不仅考虑微操作序列的长短,还要考虑每个微操作的繁简,微操作的时序是由专用的应答线路控制的,即控制器发出某一微操作控制信号后,等待执行部件完成该操作后发回的“回答”或“终止”信号,才开始下一个微操作。控制线路分散并且复杂。联合控制方式是以上两种方式的结合,对不同指令的不同微操作序列及其中的每个微操作实行部分统一、部分区别对待的方法,把各种指令的不同微操作序列中那些可以公用的部分安排在一个具有固定周期节拍和严格时钟同步的时序信号下执行,把难于公用的微操作作用异步控制通过应答方式与公用部分连接起来。

5.12 什么是数据寻址? 什么是指令寻址?

解：数据寻址是寻找操作数地址的方法,指令寻址是在主存中找到将要执行的指令地址。

5.13 RISC 思想主要基于_____。

- A. 减少指令的平均执行周期数
- B. 减少硬件的复杂度
- C. 便于编译器编译
- D. 减少指令的复杂度

解：B。

5.14 下列关于 RISC 的说法中错误的是_____。

- A. RISC 的指令条数比 CISC 少
- B. RISC 的指令平均字长比 CISC 指令的平均字长短
- C. 对于大多数计算任务来说,RISC 程序所用的指令比 CISC 程序少
- D. RISC 和 CISC 都在发展

解：B。

5.15 试说明机器指令和微指令的关系。

解：一条机器指令在执行时,需要计算机做很多微操作。在微操作控制器中,一条机器指令需要由一组微指令组成的微程序来完成。因此一条指令对应多条微指令,而一条微指令可为多个机器指令服务。

5.16 微程序控制器中机器指令与微指令的关系是_____。

- A. 每一条机器指令由一条微指令来执行
- B. 每一条机器指令由一段用微指令编成的微程序来解释执行
- C. 一段机器指令组成的程序可由一条微指令来执行
- D. 一条微指令由若干条机器指令组成

解：选 B。原因参见 16 题答案。

5.17 微程序控制器主要由_____,_____,_____ 3 大部分组成,其中_____是只读型存储器,用来存放_____。

解：控制存储器,微指令寄存器,微地址形成及译码部件,控制存储器,微程序。

5.18 设有一台模型机能够执行表 5.3 所列的 7 条指令。

试用微程序方法设计该模型机的操作控制器,并画出该模型机的微程序控制逻辑框图。

解：第一步,设计其指令操作的流程图,如图 5.2 所示。

表 5.3 一台模型机的指令

指令助记符	指令功能及操作内容
LDA X	$(X) \rightarrow AC$, 存储单元 X 的内容送累加器 AC
STA X	$(AC) \rightarrow X$, 累加器 AC 的内容送存储单元 X
ADD X	$(AC) + (X) \rightarrow AC$, AC 的内容与 X 的内容(补码)加, 结果送 AC
AND X	$(AC) \wedge (X) \rightarrow AC$, AC 的内容与 X 的内容逻辑与, 结果送 AC
JMP X	$(X) \rightarrow IP$, 无条件转移
JMP Z	如果 $AC=0$, 则 $(X) \rightarrow IP$, 无条件转移
COM	$(AC) \rightarrow AC$, 累加器内容求反

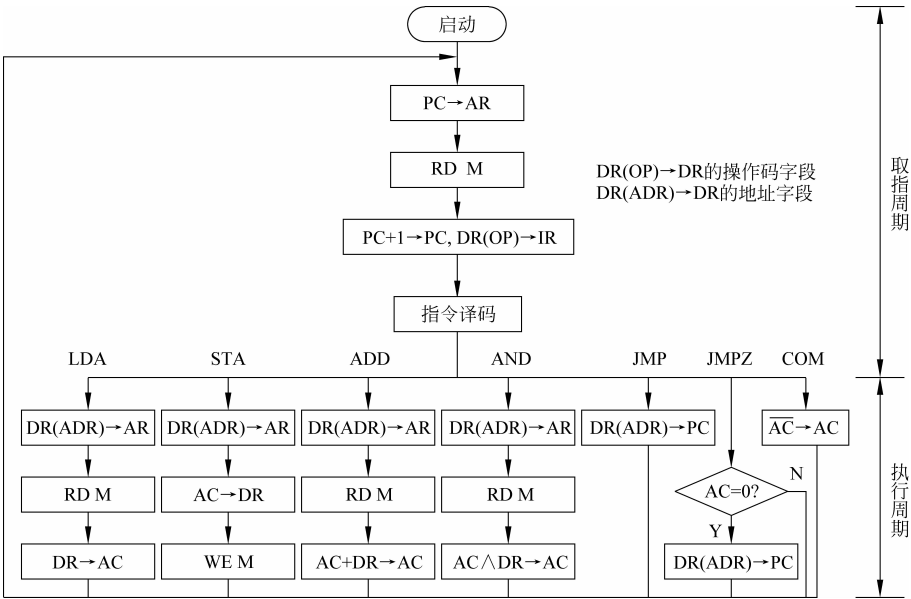


图 5.2 模拟机指令操作流程

使用的微操作如下：

```
1  IP→AR
2  RD
3  PC+1→PC
4  DR(OP)→IR
5  DR(OP)→MPC
6  DR→AC
7  AC→DR
8  WEM
9  AC+DR→AC
10 AC∧AR→AC
11 AC→AC
12 DR(ADR)→PC
13 无操作
14 DR(ADR)→AR
```

第二步,综合各种微指令格式。由上一步可知,共需 14 种微操作。
第三步,设计微指令格式。由于微操作数目不多,为简便起见,采用水平指令格式,后继