**二进制后带B,十六进制后带H,默认十进制**

**==============第二章:8086=============**

**逻辑地址**表示 段地址:偏移地址(均为16位)

**物理地址**=段地址\*16+偏移地址 (0000H~FFFFH

寄存器:用于存数据/地址偏移量/控制信息

**--------------数据寄存器-----------------**

**某x**可拆分为**某H**(高8位)和**某L**(低8位)

**AX(累加器)**:算术逻辑运算主用,也用于I/O操作.

**BX(基址寄存器)**:存放基址地址,用于存储器寻址.

**CX(计数器)**:用于循环或移位指令的计数.

**DX(数据寄存器)**:存放I/O端口地址或字乘除法的高16位结果

--下两个指针和SS段寄存器配合完成操作---

**SP(堆栈指针):**指向栈顶地址,用于堆栈操作.

**BP(基址指针):**,指向堆栈段中的特定存储单元.

**SI(源变址寄存器):**用于字符串操作或存储器寻址中的源地址偏移量.

**DI(目的变址寄存器):**用于字符串操作或存储器寻址中的目的地址偏移量.

**CS**:指向的起始地址,存放程序指令.

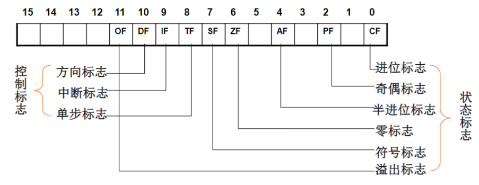
**DS:**指向数据段的起始地址,存放程序数据.

**SS**:指向堆栈段的起始地址,存放堆栈数据.

**ES(附加段寄存器):**指向**附加段**的起始地址,用于字符串操作或扩展数据存储.

**IP(指令指针寄存器)**:存放中当前指令的下一条指令的偏移地址,与CS组合形成物理地址,控制程序执行流程.

**--------------标志寄存器----------------**



**CF(进位标志)**:最高位(D15或D7)产生进位或借位时置1,否则置0.

**PF(奇偶标志):**运算结果的低8位中“1”的个数为偶数时置1,否则置0.

**AF(辅助进位标志):**低4位向高4位(D3向D4)有进位或借位时置1,否则置0.

**ZF(零标志):**运算结果为0时置1,否则置0.

**SF(符号标志):**运算结果的最高位(D15或D7)为1时置1(表示负数),否则置0.

**OF(溢出标志):**带符号数运算结果超出范围(如8位:-128~+127)时置1,否则置0.

**DF(方向标志):**控制字符串操作时地址指针的增减方向.DF=1时地址自动减量(从高到低),DF=0时自动增量(从低到高).

**IF(中断允许标志)**:控制CPU是否响应可屏蔽中断请求.IF=1时允许响应,IF=0时屏蔽中断.

**TF(单步标志)**:TF=1时CPU进入单步执行模式,每执行一条指令后暂停,便于跟踪调试.

**=============第3/4章:汇编============**

**立即寻址**:立即数只能作为源操作数,给寄存器赋初值,且源操作数和目的操作数类型要相互匹配

MOV AX,3102H ; 执行后 AX = 3102H

MOV AL,'A' ; 执行后 AL = 41H(ASCII)

**寄存器寻址**:操作数存在某寄存器中,源操作数与目的操作数**字长**需相同(匹配).操作数在寄存器reg中,无需访问总线,速度快,若选AX时间更短.

MOV AX, BX ; BX中的值传到AX中

MOV [3F00H], AX ;AX中的值传到DS\*16+3F00H

**存储器直接寻址**:

MOV AX,[2050H] ; 默认段寄存器为DS,操作数的物理地址为 DS×16 + 2050H

MOV SI,BUF ; 偏移地址以变量形式出现

MOV DX,ES:[2050H] ; 使用段超越前缀,操作数的物理地址为 ES×16 + 2050H

**寄存器间接寻址**:偏移地址存在只有SI/DI/BX/BP可做的**间址寄存器**.当使用 BP作为间址寄存器时,段寄存器使用SS.也可使用段超越.

**寄存器相对寻址**: EA = 间址寄存器+ 8/16位位移量.操作数的书写形式多样.

MOV AX,BUF[BX] ;等价于MOV AX,[BX + BUF]

**基址变址寻址**:操作数的偏移地址如下

EA=基址寄存器(BX/BP)+变址寄存器(SI/DI).

MOV AX,[BX][SI] ;等价于 MOV AX,[BX + SI]

**基址变址相对寻址:**EA=基址寄存器(BX/BP)+变址寄存器(SI/DI)+8/16位位移量.

MOV DX,disp[BX][SI] ;=MOV DX,disp[BX+SI]

**----------------基本指令-----------------**

**MOV** dest,src,将src传送给dest,源操作数内容不变,不影响标志位.

**XCHG** reg,mem/reg,内容相互交换,不影响标志位,交换只能在通用寄存器之间或通用寄存器与存储器之间进行.

**PUSH** r16/m16/seg,操作是 SP = SP - 2 ,SS:[SP] = r16/m16/seg；

**POP** r16/m16/seg,操作是 r16/m16/seg = SS:[SP] ,SP = SP + 2.

**LEA** r16,mem,将偏移地址传给指定16位寄存器.

**LAHF**:将标志寄存器FLAGS的低8位传送到AH

**SAHF**:将AH的内容传送到标志寄存器FLAGS的低8位

**PUSHF**:将标志寄存器FLAGS的内容压入堆栈

**POPF**:从堆栈中弹出数据到标志寄存器FLAGS

**XLAT**:将AL与BX寄存器的内容之和作为偏移地址,对应存储单元内容送入AL

**ADD** dest,src:dest = dest + src,影响标志位

**ADC** dest,src:dest = dest + src +CF,多精度加法

**INC** op:op = op + 1,不影响 CF

**SUB** dest,src:dest = dest - src,影响标志位

**SBB** dest,src:dest = dest - src - CF,多精度减法

**DEC** op:op = op - 1,不影响 CF

**NEG** op:op = 0 - op(取补)

**CMP** dest,src:dest - src,仅影响标志位

乘法：字节:AX=AL×src；字:DX.AX=AX×src

**MUL** src:无符号乘法, **IMUL** src:有符号乘法

除法:字节:AL=商,AH=余数；字:AX=商,DX=余数

**DIV** src:无符号除法, **IDIV** src:有符号除法,

**CBW**:将 AL 的符号扩展至 AH

**CWD**:将 AX 的符号扩展至 DX

**AND** dest,src:按位与,结果送 dest,可屏蔽某些位

**OR** dest,src:按位或,结果送 dest,可置位某些位

**XOR** dest,src:按位异或,结果送dest,可清0或求反某些位

**NOT** op:按位取反,不影响标志位

**TEST** dest,src:按位与,仅影响标志位,条件检测

**SHL** op,count:逻辑左移,最高位进CF,最低位补0

**SHR** op,count:逻辑右移,最低位进CF,最高位补0

**SAL** op,count:算术左移,同 SHL

**SAR** op,count:算术右,最低位进CF,符号位不变

**ROL** op,count:不带进位循环左移

**ROR** op,count:不带进位循环右移

**RCL** op,count:带进位循环左移

**RCR** op,count:带进位循环右移

**JMP** label:无条件转移到 label 处执行

**Jcc** label:条件 cc 成立则转移到 label

**LOOP** label:CX=CX-1,若CX≠0 则循环到 label

**LOOPZ** label:CX=CX-1,若CX≠0且ZF=1循环

**LOOPNZ** label:CX=CX-1,若CX≠0且ZF=0循环

**JCXZ** label:若 CX=0 则转移到 label

**CALL** proc:调用子程序 **RET**:子程序返回

**INT** n:产生中断类型号为 n 的内部中断

**INTO**:产生溢出中断 **IRET**:中断返回

**CLC**:CF = 0(清除进位) **STC**:CF = 1(置位进位)

**CMC**:CF 取反

**CLD**:DF = 0(清除方向标志,串操作递增)

**STD**:DF = 1(置位方向标志,串操作递减)

**CLI**:IF = 0(清除中断标志,关中断)

**STI**:IF = 1(置位中断标志,开中断)

**NOP**：空操作,消耗 3 个时钟周期

**EQU**：为常量或表达式定义符号名,不可重复

**=**：功能类似 EQU，但允许重新定义。

EMP = 5 ; 初始值5

EMP = 8 ; 重新定义为8

DB/DW/DD:定义字节/字/双字变量,分配存储单元(在DATA SEGMENT段定义)

DUP:批量初始化存储单元。

ARRAY DB 100 DUP(0) ; 定义100个字节的0**系统功能调用**(DOS 中断)

通过 INT 21H 调用 DOS 功能,常用功能如下:

显示字符(2 号功能) 显示字符串(9 号功能)

DATA SEGMENT

STR DB 'Hello!', '$' ; 字符串以'$'结尾

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

START:

MOV AX, DATA

MOV DS, AX

LEA DX, STR ; 加载字符串首地址

MOV AH, 9 ; 功能号9

INT 21H ; 显示字符串

MOV AH, 4CH ; 退出程序

INT 21H

CODE ENDS

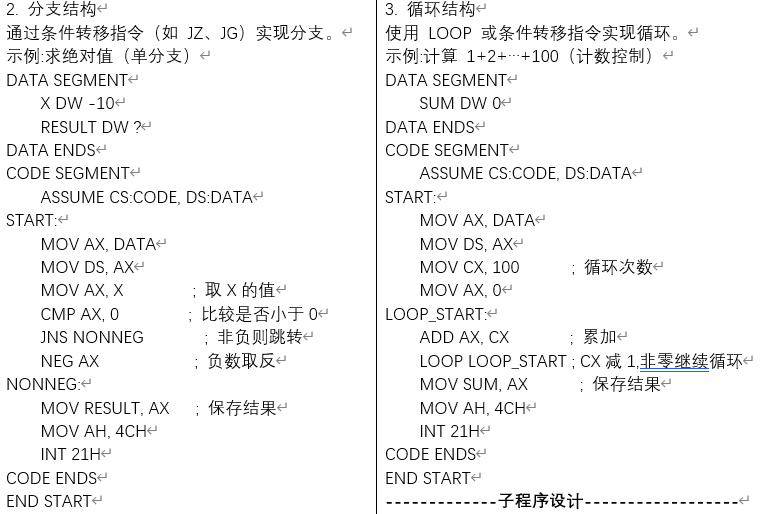
END START

键盘输入字符(1 号功能)

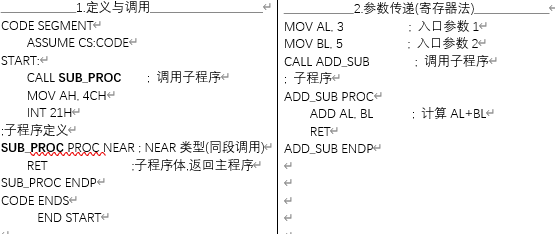
**-----------程序设计结构------------------**

1. 顺序结构

按指令顺序执行,无分支或循环。



**-------------子程序设计------------------**



**=============第7章:8259A============**

中断类型号(n):断源唯一编号(0~255),用于标识

中断向量:中断服务程序的入口地址,由段地址(CS)和偏移地址(IP)组成,共4字节。

中断向量表:存~的内存区(00000H~003FFH),共1KB.中断类型号对应的中断向量存放在地址(4\*n)处开始的4字节,偏移地址存IP(4n~4n+1),段地址存CS(4n+2~4n+3).

**-----------8259A初始化编程--------------**

ICW1写入偶地址 ICW2~ICW4写入奇地址 ICW1和ICW2必须设置 ICW3,ICW4据ICW1设置(若设置ICW3,则对主从片ICW3都设置)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ICW1\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

地址: A0=0 (端口地址为20H/A0H）

功能: 排列顺序D7->D0

D0：IC4 (1-需要ICW4，0-不需要ICW4).

D1：SNGL (1-单片模式,0-级联模式).

D2：不使用. D4：必须为 1. D5-D7:为0

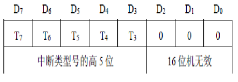
D3：LTIM (1-电平触发,0-边沿触发).

示例: OUT 20H, 13H(表示需要ICW4,单片模式,边沿触发)(13H=00010011B)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ICW2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

地址:A0=1(例如,端口地址为21H或A1H)

功能:当响应中断时,它会将IRx对应的中断类型号(T7-T3与IRx的低3位组合)发给CPU.

示例:OUT21H,08H



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ICW3\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

地址:A0=1(例如,端口地址为21H或A1H)

功能:仅在级联模式(SNGL=0)下使用.

主片ICW3：D7～D0对应中断请求输入线IR7～IR0,若接有从片,则其对应位为“1”,否则“0”。

从片ICW3：D7～D3固定为0,D2～D0为从片标志码,表示与主片的哪个中断请求输入线连接

**主片**:OUT 21H,04H(IR2引脚连接了从片).

**从片**:OUT 21H,02H(从片连接到主片的IR2引脚).

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ICW4\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

地址:A0=1(例如,端口地址为21H或A1H)

功能:设置8259A的特殊功能.D7->D0

D0：μPM(1-8086/8088模式,0-MCS-85模式).

D1：AEOI(1-自动结束中断,0-普通EOI).

D2-D4:缓冲模式设置(BUF,M/S).

D5-D7:保留，一般为0

OUT21H,01H(表示使用8086/88模式,普通EOI).

**-----------初始化示例(单片8259A,边沿触发,中断类型号基值08H,普通全嵌套,非自动结束):**

MOV AL, 00010001B ; ICW1(边沿触发,单片,需4)

OUT 20H, AL ; 偶地址20H

MOV AL,00001000B ;ICW2(中断类型号基址08H

OUT 21H, AL ;奇地址21H

MOV AL, 00000001B ;ICW4(8086模式,普通全嵌套,非自动结束) OUT 21H, AL

**---------------操作命令字----------------**

**OCW1(中断屏蔽命令字)**

写入奇地址(A0=1),每一位对应屏蔽IR引脚

MOV AL, 00101000B

OUT 21H, AL ; 屏蔽IR3、IR5

**OCW2(中断结束与优先级循环命令字)** **A0=0**

D4D3=00,特征位 D2～D0:指定L2～L0

**OCW3(特殊屏蔽与查询命令字) A0=0**

D6D5:=11,设置特殊屏蔽;=10,取消特殊屏蔽.D6=1,D5才有效 D4D3=01:特征位,

D2:中断状态查询位.若=1向8529A发中断查询命令,紧接着用IN指令读取中断查询字.

D1: =1时才允许读取D0所指定的寄存器.

D0: =1,读取ISR寄存器;=0,则读取IRR寄存器.

**------------中断向量设置----------------**

将中断服务程序的入口地址(段地址和偏移地址)写入中断向量表中的对应位置.

**直接写入中断向量表**

计算物理地址 = 中断类型号 \* 4.将中断服务程序的IP值(低两位字节)写入该物理地址. CS值(高两位字节)写入该物理地址+2的地址.

**使用DOS系统功能调用** INT 21H

获取中断向量 (INT 21H, AH=35H):

功能: 读取指定中断类型号的中断向量(CS:IP)到ES:BX寄存器对 中.

输入: AL=中断类型号 输出: ES=中断向量的段地址, BX=中断向量的偏移地址

**保护现场:** 修改系统中断向量时,程序退出前恢复原来中断向量,避免影响系统稳定性.文件中通过 PUSH ES 和 PUSH BX 来保存原中断向量,并在程序末尾通过 POP DX 和 POP DS 来恢复.

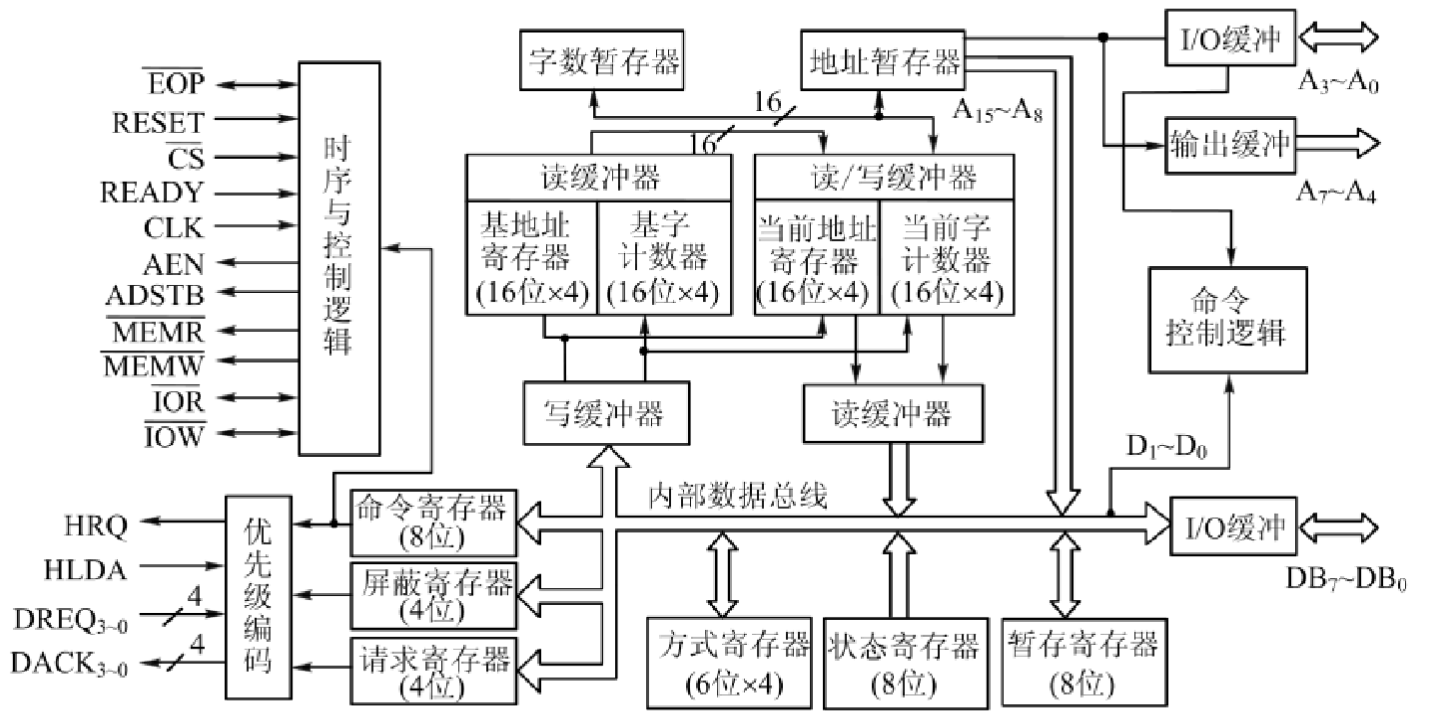
**中断服务程序的编写**:必须以IRET指令结束,而不是RET.指令在返回前会从栈中弹出标志寄存器(FLAGS)/CS/IP,确保CPU恢复到中断前的状态.

**中断屏蔽**:在设置中断向量后,还需要确保相应的中断是允许的(未被屏蔽),才能接收到中断请求.

使IMR的D0位=0,以允许IR0中断,并通过IN AL, 21H和OUT 21H, AL来操作IMR.

开中断: 最后,需要确保CPU的总中断是开放的(通过STI),这样CPU才能响应外部中断请求.

**=======第10章8237A DMA控制器==========**



**通道**:包含4个独立通道,每个通道都能够寻址64KB内存并传输多达64KB数据,**四种传输方式**

**单字节传输**:每次请求传输一个字节.**数据块传输**:传输整个数据块,直到字节计数为零或出现过程结束(EOP)信号.**请求传输**:DREQ信号有效/直到TC(终端计数)或EOP发生,传输持续.无传输请求时,CPU可用总线.**级联传输**:连接多个8237A控制器->额外的DMA通道.

**可编程性**:每个通道DMA请求都可以被允许/禁止.优先级可以是固定/循环的.

**EOP信号**:数据传输完成后生成过程结束信号.外部EOP信号也可以终止正在进行的DMA传输.

**自动初始化**:通道可编程为自动初始,EOP信号后地址/计数寄存器恢复到原始值,无需CPU干预.

**空闲循环(从属模式)**

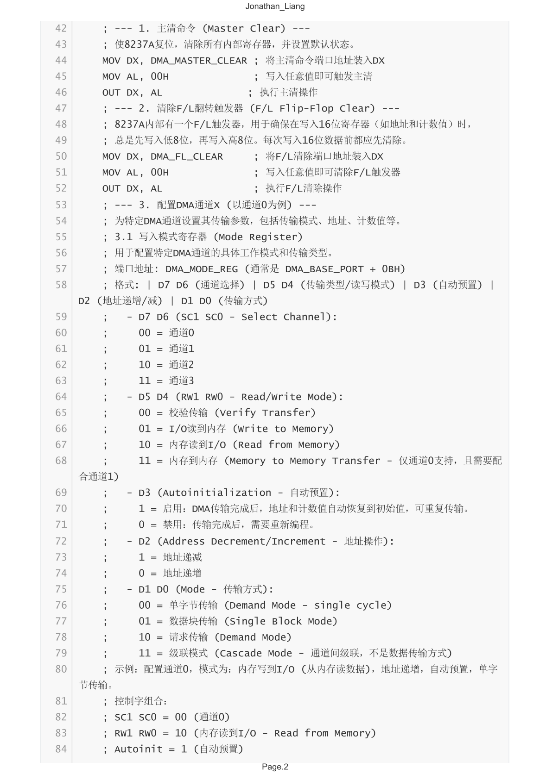
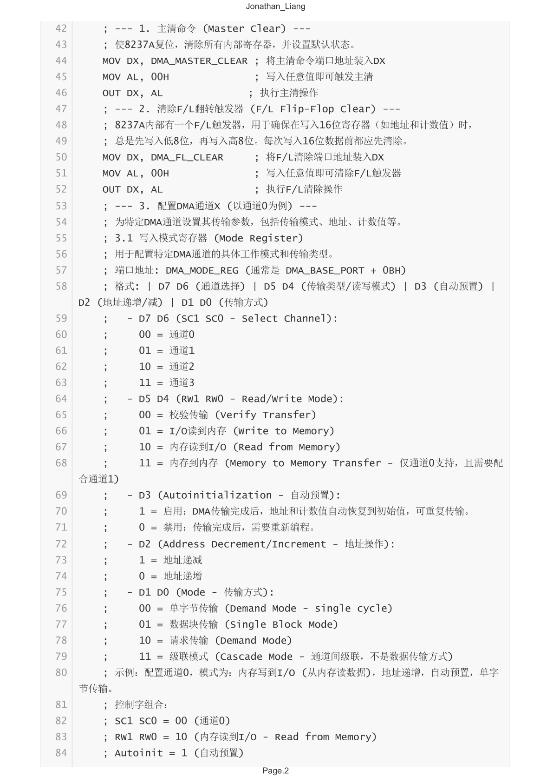
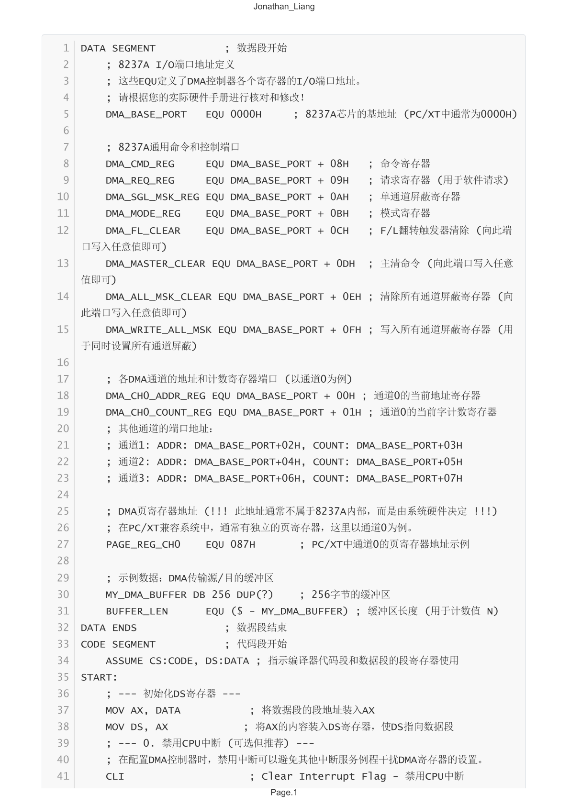
CPU通过写入其内部寄存器(指定通道/传输模式/内存起始地址/传输字节数/ 地址递增/递减等)来对8237A进行编程.可读取8237A的状态.

**活动循环(主模式)**

获得系统总线的控制权时(在CPU通过HLDA响应其HOLD请求后),它成为总线主设备.

然后,直接控制I/O设备和存储器之间或存储器与存储器之间的数据传输,无需CPU干预.

**------------8237A编程和初始化-----------**



**=============第9章:8253============**

; 8253的I/O端口地址定义

DMA\_8253\_CTR\_REG EQU 0646H ;控制字寄存器

DMA\_8253\_CNT0\_REG EQU 0640H ; 计数器0

DMA\_8253\_CNT1\_REG EQU 0642H ; 计数器1

D7 D6 D5 D4 D3 D2 D1 D0

SC1 SC0 RW1 RW0 M2 M1 M0 BCD

**SC1, SC0 (计数器选择位):**

00:选择计数器0 01:选择计1 10:选择计2

11:读回命令(用于读取计数器当前值和状态，不是常规编程)

**RW1, RW0:定义读写16位计数值**

00:锁存计数器,将计数器当前值锁存到输出寄存器，以便CPU读取。

01:只读/写低8位 10:只读/写高8位

11:先读/写低8位，后读/写高8位

通常在初始化时采用此模式

**M2, M1, M0 (工作方式选择位):工作方式。**

000 (0H):方式0 - 中断结束计数

001 (1H):方式1 - 可编程单脉冲发生器

010 (2H):方式2 - 速率发生器

011 (3H):方式3 - 方波发生器

100 (4H):方式4 - 软件触发选通

101 (5H):方式5 - 硬件触发选通

110 (6H):等同于方式2 111 (7H):等同于方式3

**BCD (计数模式选择位):**

0:二进制计数,计数值范围0到 2^16-1(65535)

1:BCD码计数,计数值范围0到 9999。

**----------------写入计数值 ------------**

8253的初始化编程分两步，第一步向控制字端口写入控制字,第二步向指定计数通道写入 计数初值

**-------完整的初始化程序结构--------------**

DATA SEGMENT ; 数据段开始

; 8253的I/O端口地址定义

DATA ENDS ; 数据段结束

CODE SEGMENT ; 代码段开始

ASSUME CS:CODE, DS:DATA

START:

; --- 初始化DS寄存器 ---

MOV AX, DATA

MOV DS, AX

; --- 1. 配置计数器0 ---

;计数器0先低后高 方式3 二进制计数BCD=0)

MOV DX, DMA\_8253\_CTR\_REG ; 将控制字寄存器端口地址装入DX

MOV AL, 36H ; 将控制字 (36H) 装入AL

OUT DX, AL;将控制字输出到控制字寄存器

; 写入计数器0的计数值：1000 (03E8H)

MOV DX, DMA\_8253\_CNT0\_REG ; 将计数器0端口地址装入DX

MOV AL, 0E8H

OUT DX, AL

MOV AL, 03H

OUT DX, AL

; --- 2. 配置计数器1 ---

计数器1先低后高方式3 二进制计数

; 组合起来是 0111 0110B = 76H

MOV DX, DMA\_8253\_CTR\_REG ; 将控制字寄存器端口地址装入DX

MOV AL, 76H

OUT DX, AL

MOV AL, 0E8H

OUT DX, AL

MOV AL, 03H

OUT DX, AL

; --- 程序退出 ---

MOV AH, 4CH

INT 21H

CODE ENDS

END START

**大写转小写**

