习　题

第一章　微型计算机概述

1.1 微处理器、微型计算机和微型计算机系统三者之间有什么不同？

1.2 CPU 在内部结构上由哪几部分组成？CPU 应具备什么功能？

1.3 累加器和其他通用寄存器相比，有何不同？

1.4 微处理器的控制信号有哪两类？

1.5 微型计算机采用总线结构有什么优点？

1.6 数据总线和地址总线在结构上有什么不同之处？如果一个系统的数据和地址合用一套总线或者合用部分总线，那么，要靠什么来区分地址和数据？

1.7 控制总线传输的信号大致有哪几种？

第二章　8086 微处理器

2.1 总线接口部件有哪些功能？请逐一进行说明。

2.2 8086 的总线接口部件由哪几部分组成？

2.3 段寄存器 CS＝1200H，指令指针寄存器 IP＝FF00H，此时，指令的物理地址为多少？指向这一物理地址的 CS 值和 IP 值是唯一的吗？

2.4 8086 的执行部件有什么功能？由哪几部分组成？

2.5 状态标志和控制标志有何不同？程序中是怎样利用这两类标志的？8086 的状态标志和控制标志分别有哪些？

2.6 8086/8088 和传统的计算机相比在执行指令方面有什么不同？这样的设计思想有什么优点？

2.7 总线周期的含义是什么？8086/8088 的基本总线周期由几个时钟组成？如一个 CPU 的时钟频率为 24MHz，那么，它的一个时钟周期为多少？一个基本总线周期为多少？如主频为 15MHz 呢？

2.8 在总线周期的 T1、T2、T3、T4 状态，CPU 分别执行什么动作？什么情况下需要插入等待状态 TW？TW 在哪儿插入？怎样插入？

2.9 从引腿信号上看，8086 和 8088 有什么不同？

2.10 在对存储器和 I/O 设备读写时，要用到$\overline{IOR}$、$\overline{IOW}$、$\overline{MR}$、$\overline{MW}$信号，这些信号在最大模式和最小模式时分别可用怎样的电路得到？请画出示意图。

2.11 CPU 启动时，有哪些特征？对 8086/8088 系统的启动程序应如何去寻找？

2.12 CPU 在 8086 的微机系统中，为什么常用 AD0 作为低 8 位数据的选通信号？

2.13 8086 和 8088 在最大模式和最小模式时，引腿信号分别有什么不同？

2.14 8086 和 8088 是怎样解决地址线和数据线的复用问题的？ALE 信号何时处于有效电平？

2.15 $\overline{BHE}$信号和 A0 信号是通过怎样的组合解决存储器和外设端口的读/写的？这种组合决定了 8086 系统中存储器偶地址体及奇地址体之间应该用什么信号区分？怎样区分？

2.16 RESET 信号来到后，CPU 的状态有哪些特点？

2.17 在中断响应过程中，8086 往 8259A 发的两个$\overline{INTA}$信号分别起什么作用？

2.18 总线保持过程是怎样产生和结束的？画出时序图。

2.19 8086 系统在最小模式时应该怎样配置？不看书画出这种配置并标出主要信号的连接关系。

2.20 时钟发生器的功能是什么？画出它的线路图。

2.21 8086 在最大模式下应当怎样配置？最大模式时为什么一定要用总线控制器？总线

控制器的输入信号是什么？输出信号是什么？

2.22 在编写程序时，为什么通常总要用开放中断指令来设置中断允许标志？

2.23 T1 状态下，数据/地址线上是什么信息？用哪个信号将此信息锁存起来？数据信息是在什么时候给出的？用时序图表示出来。

2.24 画出 8086 最小模式时的读周期时序。

2.25 8086 最多可有多少级中断？按照产生中断的方法分为哪两大类？

2.26 非屏蔽中断有什么特点？可屏蔽中断有什么特点？分别用在什么场合？

2.27 什么叫中断向量？它放在哪里？对应于 1CH 的中断向量存放在哪里？如果 1CH 的中断处理子程序从 5110：2030 开始，则中断向量应怎样存放？

2.28 从 8086/8088 的中断向量表中可以看到，如果一个用户想定义某个中断，应该选择在什么范围？

2.29 非屏蔽中断处理程序的入口地址怎样寻找？

2.30 叙述可屏蔽中断的响应过程，一个可屏蔽中断或者非屏蔽中断响应后，堆栈顶部四个单元中为什么内容？

2.31 一个可屏蔽中断请求来到时，通常只要中断允许标志为 1，便可在执行完当前指令后响应，在哪些情况下有例外？

2.32 在对堆栈指针进行修改时，要特别注意什么问题？为什么？

2.33 在编写中断处理子程序时，为什么要在子程序中保护许多寄存器？有些寄存器即使在中断子程序中并没有用到也需要保护，这又是为什么（联系串操作指令执行时遇到中断这种情况来回答）？

2.34 一个可屏蔽中断响应时，CPU 要执行哪些读/写周期？对一个软件中断又如何？

2.35 中断处理子程序在结构上一般是怎样一种模式？

2.36 软件中断有哪些特点？在中断处理子程序和主程序的关系上，软件中断和硬件中断

有什么不同之处?

2.37 系统中有多个总线模块时,在最大模式和最小模式下分别用什么方式来传递总线控制权?

2.38 8086的存储器空间最大为多少?怎样用16位寄存器实现对20位地址的寻址?

2.39 IBM PC/XT系统中,哪个区域为显示缓冲区?哪个区域用来存放中断向量?在FFFF0H到FFFFFH单元中存放什么内容?

第三章 8086的寻址方式和指令系统

3.1 8086汇编语言指令的寻址方式有哪几类?用哪一种寻址方式的指令执行速度最快?

3.2 直接寻址方式中,一般只指出操作数的偏移地址,那么,段地址如何确定?如果要用某个段寄存器指出段地址,指令中应如何表示?

3.3 在寄存器间接寻址方式中,如果指令中没有具体指明段寄存器,那么,段地址如何确定?

3.4 用寄存器间接寻址方式时,BX、BP、SI、DI分别针对什么情况来使用?这四个寄存器组合间接寻址时,地址是怎样计算的?举例进行说明。

3.5 设DS=2100H,SS=5200H,BX=1400H,BP=6200H,说明下面两条指令所进行的具体操作:

```
MOV   BYTE  PTR  [BP], 2000
MOV   WORD  PTR  [BX], 2000
```

3.6 使用堆栈操作指令时要注意什么问题?传送指令和交换指令在涉及内存操作数时分别要注意什么问题?

3.7 下面这些指令中哪些是正确的?哪些是错误的?如是错误的,请说明原因。

```
XCHG    CS, AX
MOV     [BX], [1000]
XCHG    BX, IP
PUSH    CS
POP     CS
IN      BX, DX
```

```
MOV     BYTE [BX], 1000
MOV     CS, [1000]
```

3.8 8086 系统中，当对 SS 和 SP 寄存器的值进行修改时，有什么特殊规定？这样做的原因是什么？

3.9 以下是格雷码的编码表

0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101

请用换码指令和其他指令设计一个程序段，实现格雷码往 ASCII 码的转换。

3.10 用加法指令设计一个简单程序，实现两个十六位十进制数的相加，结果放在被加数单元。

3.11 为什么用增量指令或减量指令设计程序时，在这类指令后面不用进位标志作为判断依据？

3.12 用乘法指令时，特别要注意先判断用有符号数乘法指令还是用无符号数乘法指令，这是为什么？

3.13 字节扩展指令和字扩展指令用在什么场合？举例说明。

3.14 什么叫 BCD 码？什么叫组合的 BCD 码？什么叫非组合的 BCD 码？8086 汇编语言在对 BCD 码进行加、减、乘、除运算时，采用什么方法？

3.15 用普通运算指令执行 BCD 码运算时，为什么要进行十进制调整？具体讲，在进行 BCD 码的加、减、乘、除运算时，程序段的什么位置必须加上十进制调整指令？

3.16 普通移位指令和循环移位指令(带 CF 的和不带 CF 的两类)在执行操作时，有什么差别？在编制乘除法程序时，为什么常用移位指令来代替乘除法指令？试编制一个程序段，实现将 BX 中的数除以 10，结果仍放在 BX 中。

3.17 串操作指令使用时，特别要注意和 SI、DI 这两个寄存器及方向标志 DF 密切相关。请具体就指令 MOVSB/MOVSW、CMPSB/CMPSW、SCASB/SCASW、LODSB/LODSW、STOSB/STOSW 列表说明和 SI、DI 及 DF 的关系。

3.18 用串操作指令设计实现如下功能的程序段：首先将 100H 个数从 2170H 处搬到 1000H 处，然后，从中检索相等于 AL 中字符的单元，并将此单元值换成空格符。

3.19 在使用条件转移指令时，特别要注意它们均为相对转移指令，请解释“相对转移”的含义。如果要往较远的地方进行条件转移，那么，应该怎样做？

3.20 带参数的返回指令用在什么场合？设栈顶地址为 3000H，当执行 RET 0006 后，SP 的值为多少？

3.21 用循环控制指令设计程序段，从 60H 个元素中寻找一个最大值，结果放在 AL 中。

3.22 中断指令执行时，堆栈的内容有什么变化？中断处理子程序的入口地址是怎样得到的？

3.23 中断返回指令 IRET 和普通子程序返回指令 RET 在执行时，具体操作内容有什么不同？

3.24 断点中断是指怎样一种中断？在程序调试中有什么作用？断点中断指令有什么特点？设置断点过程对应了一种什么操作？这种操作会产生什么运行结果？

3.25 HLT 指令用在什么场合？如 CPU 在执行 HLT 指令时遇到硬件中断并返回后，以下应执行哪条指令？

3.26 总线封锁指令用在什么场合？以飞机订票系统为例说明总线封锁指令的作用（设飞机订票系统为一个多处理器系统，每个处理器都是平等的）。

3.27 设当前 SS＝2010H，SP＝FE00H，BX＝3457H，计算当前栈顶地址为多少？当执行 PUSH BX 后，栈顶地址和栈顶二个字节的内容分别是什么？

3.28 在 DS 段中有一个从 TABLE 开始的 160 个字符组成的链表，设计一个程序，实现对此表进行搜索，找到第一个非 0 元素后，将此单元和下一单元清 0。

3.29 下面的程序段将 ASCII 码的空格字符填满 100 个字节的字符表。阅读这一程序

段，画出流程，并说明使用 LCD 指令和 REP STOSB 指令的作用，再指出 REP STOSB 指令使用时和哪几个寄存器的设置有关？

```
        MOV     CX, SEG TABLE          ;TABLE 为字节表表头
        MOV     ES, CX
        MOV     DI, OFFSET TABLE       ;DI 指向字节表
        MOV     AL,' '                 ;空格符送 AL
        MOV     CX,64H                 ;字节数
        CALL    FILLM                  ;调用填数子程序
        ⋮                              ;后续处理
        ⋮
FILLM:  JCXZ    EXIT                   ;CX 为 0 则退出
        PUSH    DI                     ;保存寄存器
        PUSH    CX
        CLD                            ;方向标志设置
        REP     STOSB                  ;重复填数
        POP     CX
        POP     DI
EXIT:   RET
```

3.30 以下程序段将一个存储块的内容复制到另一个存储块，进入存储段时，SI 中为源区起始地址的偏移量，DI 中为目的区起始地址的偏移量，CX 中为复制的字节数。阅读此程序段并具体说明 REP MOVSB 指令使用时与哪些寄存器有关？

```
        PUSH    DI                     ;保存寄存器
        PUSH    SI
        PUSH    CX
        CMP     DI,SI                  ;看源区和目的区的地址哪个高
        JBE     LOWER                  ;如目的区地址较低，则转移
        STD                            ;目的区地址高，则设方向标志为 1
        ADD     SI,CX                  ;从最后一个字节开始复制
        DEC     SI                     ;调整源区地址
        ADD     DI,CX
        DEC     DI                     ;调整目的区地址
        JMP     MOVEM
LOWER:  CLD                            ;从第一个字节开始复制
MOVEM:  REP     MOVSB
        POP     CX
        POP     SI
        POP     DI
        RET
```

3.31 下面的程序段实现对两个存储区中的字进行比较。如找到一对不同的字，则退出，此时，ZF 标志为 0，DI 指向此字；如两个存储块中所有字均一一相同，则退出程序时，CX 中值为 0，ZF 标志为 1。阅读这一程序段，并仿此设计一个比较字节块的程序段。

```
MATT:   MOV     SI,OFFSET SOURCE        ;源区首址
        MOV     DI,OFFSET TARGET        ;目的区首址
        MOV     CX,NUMBER
        JCXZ    EUIT                    ;如 CX 为 0,则结束
        PUSH    CX                      ;保存有关寄存器
        PUSH    SI
        PUSH    DI
        CLD                             ;清方向标志
        REPE    CMPSW                   ;比较
        JZ      MATCH                   ;ZF 标志为 1,则转移
        PUSHF                           ;ZF 标志为 0,则 DI 指向此字
        SUB     DI,2
        POPF
        JMP     EXIT                    ;再退出
MATCH:  POP     DI                      ;恢复寄存器
        POP     SI
        POP     CX
EXIT:   RET
```

3.32 下面的程序段实现。在 TABLE 为起始地址的 100 个字符长度的表中检索“$”字符请分析这一程序段，然后说明 REPNE SCASB 指令的具体执行过程。

```
START:  MOV     CX,SEG TABLE            ;表段地址送 ES
        MOV     ES,CX
        MOV     DI,OFFSET TABLE         ;表偏移量送 DI
        MOV     AL,'$'                  ;检索的关键字
        MOV     CX,64H                  ;检索的字节数
        PUSH    DI                      ;保存起始地址
        CLD                             ;清除方向标志
        REPNE   SCASB                   ;检索
        JNZ     NFOUN                   ;如未找到,则转移
        SUB     DI,1                    ;找到,则指向此字符
        JMP     EXIT
NFOUN:  POP     DI                      ;恢复起始地址
EXIT:   RET
```

3.33 下面的程序段实现两个 32 位不带符号数的相乘，被乘数在 DX 和 AX 寄存器中，乘数在 CX 和 BX 寄存器中，最后的 64 位乘积在 DX，CX，BX，AX 中。图 1 说明乘

法过程。读懂程序段和附图，并自己设计一个程序，实现一个 16 位数和一个 32 位非符号数相乘。

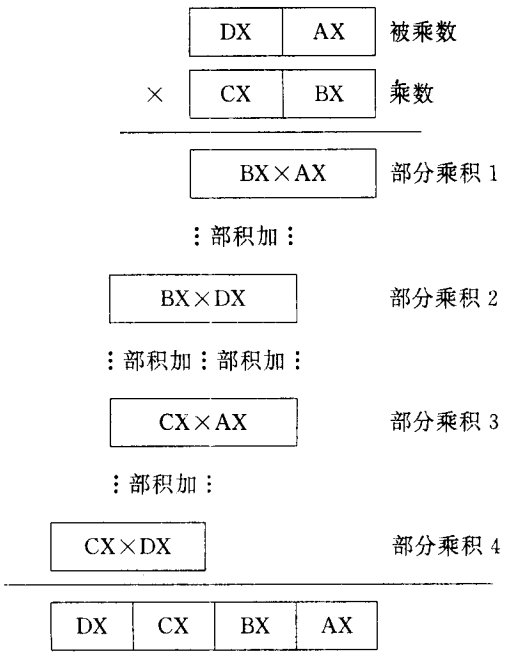

图 1　两个 32 位非符号数相乘流程图

```
STAT:   JMP   MUL64
HI0     DW    ?
LO0     DW    ?
HI1     DW    ?
LO1     DW    ?
HI2     DW    ?
LO2     DW    ?
HI3     DW    ?
LO3     DW    ?
HI4     DW    ?
LO4     DW    ?
MUL64:  MOV   HI0,DX          ;保存被乘数
        MOV   LO0,AX
        MUL   BX              ;得部分乘积 1
        MOV   HI1,DX          ;保存部分乘积 1
        MOV   LO1,AX
        MOV   AX,HI0          ;得部分乘积 2
        MUL   BX
        MOV   HI2,DX          ;保存部分乘积 2
        MOV   LO2,AX
```

```
        MOV   AX,LO0          ;得部分乘积 3
        MUL   CX
        MOV   HI3,DX          ;保存部分乘积 3
        MOV   LO3,AX
        MOV   AX,HI0          ;得部分乘积 4
        MUL   CX
        MOV   HI4,DX          ;保存部分积 4
        MOV   LO4,AX
        MOV   AX,LO1          ;乘积的低 16 位在 AX 中
        MOV   BX,HI1          ;乘积的次低 16 位在 BX 中
        ADD   BX,LO2
        ADC   HI2,0
        ADD   BX,LO3
        MOV   CX,HI2          ;乘积的次高 16 位在 CX 中
        ADC   CX,HI3
        ADC   HI4,0
        ADD   CX,LO4
        MOV   DX,HI4          ;乘积的高 16 位在 DX 中
        ADC   DX,0
        RET
```

3.34 下面的程序实现两个 32 位带符号数的乘法，其中调用了题 3.34 中的非符号数相乘的程序 MUL64，结果放在 DX、CX、BX、AX 四个寄存器中，进入程序时，DX、AX 中为被乘数，CX、BX 中为乘数。读懂程序后请设计一个 16 位带符号数和 32 位带符号数相乘程序。

```
MULS64: MOV   [1000],0        ;1000 单元作为负数标志
        CMP   DX,0            ;被乘数为负数吗?
        JNS   CHKK            ;否，则转 CHKK
        NOT   AX              ;是，则取补码
        NOT   DX
        ADD   AX,1
        ADC   DX,0
        NOT   [1000]          ;负数标志置 1
CHKK:   CMP   CX,0            ;乘数为负数吗?
        JNS   GOMUL           ;否，则转 GOMUL
        NOT   BX              ;是，则取补码
        NOT   CX
        ADD   BX,1
        ADC   CX,0
        NOT   [1000]          ;且负数标志取反
GOMUL:  CALL  MUL64           ;调用非符号数乘法程序
```

```
        CMP     [1000],0            ;结果为正数吗?
        JZ      EXIIT               ;是正数,则转移
        NOT     AX                  ;是负数,则取补码
        NOT     BX
        NOT     CX
        NOT     DX
        ADD     AX,1
        ADC     BX,0
        ADC     CX,0
        ADC     DX,0
EXIIT:  RET
```

3.35 下面是一个实现 16 位非组合 BCD 码相加的程序段,阅读这一程序段后再设计一个实现 16 位非组合 BCD 码减法的程序。

```
ANBCD:  MOV     CH,AH               ;进入程序段时,AX 中为第二个操作数
        ADD     AL,BL               ;BX 中为被加数,实现低八位相加
        AAA
        XCHG    AL,CH
        ADC     AL,BH               ;实现高八位相加
        AAA
        MOV     AH,AL               ;和保存在 AX 中
        MOV     AL,CH
        RET
```

3.36 下面的程序段实现两个 16 位组合 BCD 码相减,进入程序时,BX 中为被减数,AX 中为减数,程序执行后,结果在 AX 中。请仿照这一程序段设计两个 16 位组合 BCD 码相加的程序。

```
STASUB: MOV     CH,AH               ;保存高八位
        SUB     AL,BL               ;低八位相减
        DAS                         ;十进制调整
        XCHG    AL,CH
        SBB     AL,BH               ;高八位相减
        DAS
        MOV     AH,AL               ;结果在 AX 中
        MOV     AL,CH
        RET
```

3.37 下面是一个实现组合的 32 位 BCD 码除以组合的 16 位 BCD 码的程序,结果得到 16 位组合的 BCD 码的商和 16 位组合的 BCD 码余数。进入程序时,被除数在 DX、AX 中,除数在 BX 中,程序执行后,商在 AX 中,余数在 DX 中。请为子程序 2BCD 加上详细注释,再分析整个程序,并画出详细流程图。

```
DIBCD:  PUSH  AX          ;被除数低 16 位进堆栈
        MOV   AX,BX       ;除数送 AX
        CALL  BCD2        ;将除数转换为二进制数
        MOV   BX,AX       ;除数送回 BX
        MOV   AX,DX       ;将被除数高 16 位转换为二进制数
        CALL  BCD2
        MOV   CX,10000    ;被除数高 16 位乘 10000
        MUL   CX
        MOV   SI,AX       ;被除数高 16 位保存到 SI
        POP   AX
        CALL  BCD2        ;被除数低 16 位转换为二进制数
        ADD   AX,SI
        ADC   DX,0        ;DX 和 AX 中得到二进制被除数
        DIV   BX          ;除法运算
        MOV   CX,AX       ;商存入 CX
        MOV   AX,DX       ;余数存入 AX
        CALL  2WBCD       ;余数转换为 BCD 码
        MOV   DX,AX       ;余数送 DX
        MOV   AX,CX       ;商转换为 BCD 码
        CALL  2WBCD
        RET
BCD2:   MOV   SI,AX
        SUB   AX,AX
        CALL  CONVER      ;转换最高一个 BCD 码(4 位二进制)
        CALL  CONVER      ;转换次高一个 BCD 码
        CALL  CONVER      ;转换次低一个 BCD 码
        CALL  CONVER      ;转换最低一个 BCD 码
        RET
CONVER: MOV   DI,0        ;清 DI
        MOV   CX,4        ;四次移位
SHIF:   SHL   SI,1        ;左移一次
        RCL   DI,1        ;左移的数位进入 DI
        LOOP  SHIF
        MOV   CX,10       ;结果乘 10
        MUL   CX
        ADD   AX,DI       ;加上新的数,结果在 AX 中
        RET
2WBCD:  CMP   AX,9999     ;数据是否太大?
        JBE   2BCD        ;否,则转 2BCD
        STC
        JC    EXIT        ;是,则退出
2BCD:   SUB   DX,DX
```

```
        MOV   CX,1000
        DIV   CX
        XCHG  AX,DX
        MOV   CL,4
        SHL   DX,CL
        MOV   CL,100
        DIV   CL
        ADD   DL,AL
        MOV   CL,4
        SHL   DX,CL
        XCHG  AL,AH
        SUB   AH,AH
        DIV   CL
        ADD   DL,AL
        MOV   CL,4
        SHL   DX,CL
        ADD   DL,AH
        MOV   AX,DX
EXIT:   RET
```

3.38 以下程序将一个八位二进制数转换为两位 BCD 数字，进入程序时，AL 中为二进制数，退出程序时，如 CF 为 0 则 AL 中为 BCD 数字，如 CF 为 1，则表示由于输入值超出范围故结果无效。阅读下面程序后，画出流程图，然后设计一个将组合的 BCD 码(两位)转换为八位二进制数的程序。

```
START:  CMP   AL,99          ;是否超出范围?
        JBE   STRAT
        STC                  ;是，则转 EXIT，并给 CF 置 1
        JC    EXIT
STRAT:  MOV   CL,10          ;10 作为除数
        XOR   AH,AH
        CBW                  ;将 AL 中数扩展到 AH
        DIV   CL             ;除法结果 AL 中为高位，AH 中为低位
        MOV   CL,4
        SHL   AL,CL          ;左移四位
        OR    AL,AH          ;合成 BCD 码在 AL 中
EXIT:   RET
```

3.39 下面程序采用 XLAT 指令将二进制数换成十六进制数，阅读下面程序，体会 XLAT 换码指令的用法，然后，设计一个查表程序，功能为按学号查找学生姓名。

```
START:  JMP   BINASC
ASCII   DB    '0123456789ABCDEF'
```

```
BINASC:  PUSH  BX
         AND   AL,0FH          ;清除 AL 中高四位
         LEA   BX,ASCII        ;BX 指向 ASCII 表
         XLAT                  ;转换为 ASCII 码
         POP   BX
         RET
```

3.40 下面的程序将两个字符串合并为一个字符串。在进入程序前，设第一个字符串的偏移量和长度已分别放在 DI 和 BX 中，第二个字符串的偏移量和长度则分别放在 SI 和 CX 中。阅读下面程序并画出流程图，在此基础上，再设计一个将三个字符串合并为一个字符串的程序(设进入程序前第三个字符串的偏移量和长度分别在 DX 和 AX 中)。

```
START:   JCXZ  EXIT           ;如第二串长度为 0,则退出
         CMP   BX,0
         JEE   EXIT           ;如第一串长度为 0,则退出
         PUSH  DI             ;保存第一串地址
         PUSH  CX
         PUSH  DI
         ADD   DI,BX          ;计算第一串末地址
         CMP   SI,DI          ;第一串末地址是否超过第二串首地址?
         JA    OKOK1           否,则转
         POP   DI
         PUSH  SI             ;保存第二串地址
         ADD   SI,CX          ;计算第二串末地址
         CMP   SI,DI          ;第二串末地址是否高于第一串首地址?
         POP   SI
         JBE   OKOK2          ;否,则转
         SUB   DI,DI          ;是,则使 ZF 为 0,且退出
         JZ    EXEX
OKOK1:   POP   DI
OKOK2:   CLD                  ;连接两个字符串
         REP   MOVSB
         MOV   SI,DI          ;SI 指向新串首址
         POP   CX
         ADD   BX,CX          ;BX 为新串长度
EXEX:    POP   CX
         POP   DI
EXIT:    RET
```

3.41 下列程序将第二个字符串插入第一个字符串中的指定位置，为此，要在插入点处将第一串的后面部分往后移动第二串长度的空间，设进入程序时，第一串的偏移量和

长度分别在DI和BX中，第二串的偏移量和长度分别在SI和CX中，BP为插入点的偏移量。阅读程序并画出流程，再说明ZF在程序中的作用，并对插入过程作详细注释。

```
START:  JMP   INSERT
SIZE2   DW    ?          ;保存CX用
OFF1    DW    ?          ;保存DI用
INSERT: JCXZ  QUIT       ;第二串为空串，则退出
        CMP   BX,0
        JE    QUIT       ;第一串为空串，则退出
        MOV   SIZE2,CX   ;保存CX
        MOV   OFF1,DI    ;保存DI
        ADD   DI,BX
        CMP   SI,DI      ;第二串是否已在第一串中?
        JAE   OKOK       ;否，则转
        PUSH  SI
        ADD   SI,CX
        CMP   SI,OFF1    ;第一串是否覆盖第二串?
        POP   SI
        JBE   OKOK       ;否，则转
        SUB   DI,DI      ;使ZF为0
        JZ    EXIT
OKOK:   STD              ;完成插入
        PUSH  SI
        DEC   DI
        MOV   SI,DI
        ADD   DI,CX
        MOV   CX,SI
        SUB   CX,BP
        INC   CX
        REP   MOVSB
        CLD
        POP   SI
        MOV   CX,SIZE2
        MOV   DI,BP
        REP   MOVSB
        MOV   SI,BP      ;SI指向新址
        ADD   BX,SIZE2   ;BX为新串长
EXIT:   MOV   DI,OFF1    ;恢复DI
        MOV   CX,SIZE2   ;恢复CX
QUIT:   RET
```

第四章　微型计算机和外设的数据传输

4.1 外部设备为什么要通过接口电路和主机系统相连?存储器需要接口电路和总线相连吗？为什么?

4.2 是不是只有串行数据形式的外设需要接口电路和主机系统连接？为什么?

4.3 接口电路的作用是什么？按功能可分为几类?

4.4 数据信息有哪几类？举例说明它们各自的含义。

4.5 CPU 和输入/输出设备之间传送的信息有哪几类?

4.6 什么叫端口？通常有哪几类端口？计算机对 I/O 端口编址时通常采用哪两种方法？在 8086/8088 系统中,用哪种方法对 I/O 端口进行编址?

4.7 为什么有时候可以使两个端口对应一个地址?

4.8 CPU 和外设之间的数据传送方式有哪几种?实际选择某种传输方式时,主要依据是什么?

4.9 无条件传送方式用在哪些场合？画出无条件传送方式的工作原理图并说明。

4.10 条件传送方式的工作原理是怎样的？主要用在什么场合？画出条件传送(查询)方式输出过程的流程图。

4.11 设一个接口的输入端口地址为 0100H,而它的状态端口地址为 0104H,状态口中第 5 位为 1 表示输入缓冲区中有一个字节准备好,可输入。设计具体程序实现查询式输入。

4.12 查询式传送方式有什么优缺点？中断方式为什么能弥补查询方式的缺点?

4.13 画一个用中断方式进行输出传输的接口电路。

4.14 叙述可屏蔽中断的响应和执行过程。

4.15 通常解决中断优先级的方法有哪几种？各有什么优缺点?

4.16 和 DMA 方式比较，中断传输方式有什么不足之处？

4.17 叙述用 DMA 方式传输单个数据的全过程。

4.18 DMA 控制器的地址线为什么是双向的？什么时候往 DMA 控制器传输地址？什么时候 DMA 控制器往地址总线传输地址？

4.19 在设计 DMA 传输程序时，要有哪些必要的程序模块？设计一个启动数据块输出的程序段。

4.20 在查询方式、中断方式和 DMA 方式中，分别用什么方法启动数据传输过程？

第五章　串并行通信和接口技术

5.1 接口部件为什么需要有寻址功能？设计一个用 74LS138 构成的译码电路，输入为 A3、A4、A5、A8，输出八个信号以对八个接口部件进行选择。想一想如果要进一步对接口中的寄存器进行寻址，应该怎样实现？

5.2 接口部件的输入/输出操作具体对应哪些功能，举例说明。

5.3 从广义上说接口部件有哪些功能？

5.4 怎样进行奇/偶校验？如果用偶校验，现在所传输的数据中 1 的个数为奇数，那么，校验位应为多少？

5.5 什么叫覆盖错误？接口部件如何反映覆盖错误？

5.6 接口部件和总线之间一般有哪些部件？它们分别完成什么功能？

5.7 为什么串行接口部件中的四个寄存器可以只用一位地址来进行区分？

5.8 在数据通信系统中，什么情况下可以采用全双工方式，什么情况下可用半双工方式？

5.9 什么叫同步通信方式？什么叫异步通信方式？它们各有什么优缺点？

5.10 什么叫波特率因子？什么叫波特率？设波特率因子为 64，波特率为 1200，时钟频率为多少？