**第一章**

地址总线：宽度决定寻址范围

数据总线：宽度决定数据传输

控制总线：宽度决定控制外部器件能力

8086cpu分配内存地址空间：

0~07FFFH:32KB空间为主随机存储器的地址空间

8000H~9FFFH:8KB空间为显存地址空间

A000H~FFFFH:24KB空间为各个ROM的地址空间

**第二章**

通用寄存器AX/BX/CX/DX，每个寄存器16位，都可以分为H和L，独立的8位。

8086CPU采用在内部用两个16位地址合成的方法来形成一个20位的物理地址。地址加法器采用**物理地址=段地址\*16+偏移地址**的方法合成物理地址。

内存没有分段，段的划分来自CPU，8086CPU中，一个段的最大长度是64KB，因为偏移地址只有16位。

8086CPU有4个段寄存器：CS/DS/SS/ES

CS和IP配合执行指令

**第三章**

DS通常用来存放要访问数据的段地址

mov、add、sub、pop、push

SS和SP配合指向栈顶元素，栈从高地址向低地址生长。

执行push时，CPU的两步操作是：先改变SP，后向SS:SP处传送。执行pop时，CPU的两步操作是：先读取SS:SP处的地址，然后改变SP。

**第四章**

伪指令没有对应的机器指令，最终不被CPU执行。伪指令是由编译器来执行的指令，编译器根据伪指令来进行相关的编译工作。

1.segment和ends的功能定义一个段：

段名 segment

内容

段名 ends

2.end是一个汇编程序的结束标记

3.assume，该指令假设某一段寄存器和程序中的某一个用segment…ends定义的段相关联。

标号指代了一个地址。

DOS系统中，.exe文件加载时，会在内存区的前256字节中创建一个称为程序段前缀(PSP)的数据区，DOS用PSP和被加载程序进行通信。

**第五章**

loop 标号

CPU执行loop指令时，进行两步操作，(CX)=(CX)-1,之后判断CX中的值，如果不为0跳转至标号处执行，为0向下执行。

DOS和其他合法的程序一般不会使用0:200~0:2FF的256字节空间。

**第六章**

dw：define word，定义字型数据

伪指令end可以通知编译器程序的入口在什么地方，比如end start说明程序入口在start标号处。

**第七章**

and，or

小写字母的ASCII码值比大写字母的ASCII码值大20H。通过or al,00100000B指令可以将字母变为小写字母。

si和di是8086CPU中和bx功能接近的寄存器，si和di不能分为两个8位寄存器使用。

**第八章**

在没有寄存器名存在的情况下，用操作符X ptr指明内存单元的长度，X在汇编指令中可以为word或byte。

div除法指令

div reg

div 内存单元

除数有8位和16位两种，在一个reg或者内存单元中，在指令中直接给出。

被除数默认放在AX或DX和AX中，如果除数为8位，被除数则是16位，默认在AX中，如果除数是16位，被除数32位，高16位在DX，低16位在AX

除数为8位，AL存储除法操作的商，AH存储除法操作的余数；除数为16位，AX存储除法操作的商，DX存储除法操作的余数

dd: define double word，定义双字型数据

dup，配合db、dw、dd等数据定义伪指令，用来进行数据的重复。使用格式:db/dw/dd 重复的次数 dup (重复的字节/字/双字型数据)

**第九章**

只修改IP时，称为段内转移，同时修改CS和IP时，称为段间转移。由于IP的修改范围不同，段内转移分为短转移和近转移。短转移的IP修改范围为-128~127；近转移的IP修改范围为-32768~32767.

8086CPU的转移指令分为：无条件转移指令，条件转移指令，循环指令，过程，中断。

操作符offset在汇编语言中由编译器处理，功能是取得标号的偏移地址。

jmp short 标号，该指令实现段内短转移。

转移指令中并没有告诉CPU要转移的目的地址，但是告诉CPU要转移的位移。这个位移，是编译器根据汇编指令中的”标号”计算出来的。

图示

AI 生成的内容可能不正确。图中可以看到，计算偏移地址是由下一个执行指令的地址和要跳转的指令地址之差组成。

jmp far ptr标号，该指令实现段间转移，又称为远转移。

**附注3**内容补充：masm汇编器对jmp的处理，编译器有一个地址计数器AC，在编译程序过程中，每读到一个字节AC就加1。遇到伪操作如db、dw时，也会根据具体情况使AC增加。

--在向前转移时，编译器读到标号s后记下AC的值，读到jmp ...s后就可以算出位移量disp。如果disp在短转移范围内，不论是哪种格式，都转变成jmp short s对应机器码。jmp short s机器码格式是EB disp(占两个字节)；如果disp在近转移范围内，使用jmp short s产生编译错误，对于jmp s、jmp near ptr s产生jmp near ptr s对应机器码。jmp near ptr s机器码格式是E9 disp(占三个字节)，对于jmp far ptr s产生机器码EA 偏移地址 段地址(占五个字节)。

--在向后转移时，未读到标号，无法确定disp大小。此时编译器将jmp ...s指令都当做jmp short s读取，记下指令位置以及AC对应值。对于jmp short s，编译器生成EB和1个nop指令(预留1字节空间放disp)；对于jmp s和jmp near ptr s，生成EB和两个nop指令，对于jmp far ptr s，编译器生成EB和4个nop指令。之后读取到标号s时，计算处disp，然后如果在短转移范围内，就直接添加jmp short s对应的机器码，所以会看到有的指令后会有多余的nop指令。如果在近转移范围内，对于jmp short s直接产生编译错误，对于jmp s和jmp near ptr s、jmp far ptr s，在前面指令位置上写上对应机器码即可。

转移地址在内存中的jmp指令有jmp word ptr 内存单元地址(段内转移)、jmp dword ptr 内存单元地址(段内转移)，高地址处的字是转移的目的段地址，低地址处是转移的目的偏移地址。

一个字符在显示缓冲区中占用两个字节，分别存放字符的ASCII码和属性。属性字节格式：

手机屏幕截图

AI 生成的内容可能不正确。

常用的黑底白字，属性字节为00000111B。

**第十章**

ret指令用栈中的数据，修改IP的内容，从而实现近转移。

retf指令用栈中的数据，修改CS和IP的内容，从而实现远转移。

call指令将当前的IP或CS和IP压入栈中，之后转移。

1. call 标号，将当前IP压栈，转到标号
2. call far ptr 标号，实现段间转移，将当前CS和IP压栈
3. call word ptr 内存单元地址
4. call dword ptr 内存单元地址

mul指令，8位相乘，一个默认放在AL中，另一个放在8位reg或者内存字节单元中；16位相乘，一个默认放在AX中，另一个放在16位reg或者内存字单元中。8位相乘结果默认放在AX中，16位相乘结果高位默认放在DX中，低位在AX存放。

附注4内容与栈帧相关，栈帧的概念在网上有较多学习界面，理解函数参数入栈、返回地址入栈，栈帧形成过程等内容。

**第十一章**

表格

AI 生成的内容可能不正确。

有的指令的执行对标志寄存器没有影响，比如mov、push、pop等，大都是传送指令。有的指令的执行对标志寄存器有影响，比如add、sub、mul、div、inc、or、and等。

ZF：结果为0时置1。

PF：所有bit位中1的个数为偶数时置1。

SF：将数据视为有符号数据后，结果为负，置1。

CF：无符号数运算，最高有效位朝更高位进位或者借位，置1。

OF：有符号数运算后，产生了溢出，置1。

DF：方向标志位，在串处理指令中，控制每次操作后si、di的增减。df为0，每次操作后递增，为1，每次操作后递减。

adc指令是带进位加法指令，利用CF上记录的进位值。该指令可以对更大数据进行加法运算(产生进位)。

sbb指令是带借位减法指令，利用CF上记录的借位值。

cmp指令是比较指令，相当于减法指令，但是不保存结果。cmp指令执行后对标志寄存器产生影响。

无符号数比较结果进行转移的条件转移指令。

图片包含 信件

AI 生成的内容可能不正确。

串传送指令：

movsb：将ds:si指向的内存单元中的字节送入es:di中，然后根据df位的值，将si和di递增或者递减1。

movsw：将ds:si指向的内存单元中的字送入es:di中，然后根据df位的值，将si和di递增或者递减2。

串传送指令可以配合rep使用，格式为：rep movsb。rep的作用是根据cx的值，重复执行后面的串传送指令。由于每执行一次movsb指令si和di都会递增或者递减指向后一个单元或者前一个单元，rep movsb可以循环实现cx个字符的传送。

8086CPU设置了cld和std指令对df位进行操作，清零或者置1。

pushf的功能是将标志寄存器的值压栈，popf的功能是从栈中弹出数据，送入标志寄存器中。

**第十二章**

中断向量表在内存中存放，对于8086PC机，中断向量表指定放在内存地址0处。从0000:0000~0000:03FF的1024个单元中存放中断向量表，一个表项占两个字，高地址存放段地址，低地址存放偏移地址。

中断处理程序的编写方法：

1. 保存用到的寄存器
2. 处理中断
3. 恢复用到的寄存器
4. iret指令返回

iret指令相当于：pop IP、pop CS、popf。

在中断过程中，寄存器入栈的顺序是标志寄存器，CS，IP，iret出栈顺序相反。

CPU在执行完一条指令之后，如果检测到标志寄存器TF位为1，则产生单步中断，引发中断过程。在进入到中断处理程序之前，设置TF=0，避免CPU在执行中断处理程序时发生单步中断。

**第十三章**

int n，n为中断码类型，功能是引发中断过程。

图片包含 文本

AI 生成的内容可能不正确。