第 1 章　微型计算机基础知识

学习要点

- 二进制、十进制、十六进制数相互转换方法
- 数据在内存储器中的存储格式
- 原码、补码（重点）、溢出（双高位判别法）
- 计算机中的数据类型：整数（有符号和无符号）、BCD 码、ASCII 码

第 2 章　微型计算机组成及微处理器功能结构

学习要点

- 微型计算机的组成及工作原理(CPU、M、I/O、外设)
- CPU 的内部结构，以及物理地址与逻辑地址
- 编程模块（寄存器阵列）（通用寄存器组、FLAG 寄存器标志位）
- 堆栈

第 3 章　80x86 寻址方式与指令系统

学习要点

- 数据寻址的 7 种寻址方式，I/O 端口寻址的 2 种寻址方式（操作数在何地、书写规则）
- 程序转移地址的 4 种寻址方式
- 掌握 8086 指令（操作码助记符，可以使用的寻址方式）（常用的要知道）

第 4 章　汇编语言程序设计

学习要点

- 掌握汇编语言源程序的框架结构，并熟悉常用伪指令。
- 掌握变量、地址标号的定义方法及属性，以及分析运算符和类型指定运算符。
- 会编写汇编程序（例题、作业）
- 掌握宏定义与调用方法，并比较宏与子程序。

第 5 章　　微处理器外部结构和总线操作时序

- 微处理器级总线与系统总线
- 总线的三态性和分时复用性
- 8086CPU 中特殊的存储体结构
- 总线周期概念理解：知道某条指令执行所需周期（读、写），知道哪些信号译码，会有何控制信号

第 6 章　半导体存储器及接口

学习要点

- 掌握存储器的分类和分级（ROM、RAM、DRAM、SRAM 概念）
- 掌握单片存储器容量与其地址线和数据线的关系（地址译码、容量怎么组成）
- 掌握静态芯片组的设计（给定内存空间，要求计算所需的芯片组、芯片数，芯片组的连接）
- 高速缓冲存储器 Cache 概念及其作用、了解基本工作原理

第 7 章　存储器管理

学习要点

- 掌握实方式存储器管理：分段结构（逻辑地址——物理地址）
- 掌握保护方式存储器管理：段页式两级存储器管理（逻辑地址——线性地址——物理地址）

第8章 中断和异常

学习要点

- 掌握中断的概念及其分类，搞清中断与子程序的区别

- 掌握 80X86 中断及异常的优先级，分清中断优先级与中断优先次序
- 重点掌握实方式下的中断（中断类型码、中断矢量、中断矢量表、中断矢量的设置、中断响应过程）
- 掌握可编程中断控制器 8259A 的编程（对 ICW、OCW 的理解：格式、用途），了解工作方式（重点在完全嵌套方式和特殊屏蔽方式）；8259A 中断级情况

第9章 输入/输出方法及常用的接口电路

学习要点

- 接口、端口概念、作用
- I/O 端口编址方法（统一、独立，I/O 接口与 8086 相连时奇、偶地址及连续地址接法）；
  基本输入/输出法：三种；指令搞清书写方法规定：IN、OUT
- 地址译码
- 8255A 并行接口（结构、引脚——硬件连线，端口地址；控制字——软件编程、理解工作方式；ISR 程序 IN、OUT；书上程序仔细阅读、理解）
- 8253 定时器（重点方式 0、1、2、3——用途、功能；控制字、计数初值设置，$N=f_{clki} \div f_{outi}$ ；CLK、GATE、OUT 的理解）

第 1 章　微型计算机中的数据类型

重要概念

1．机器数、真值

机器数：机器中符号位和数值位一起完整地表示有符号二进制数的形式称为机器数。

真　值：机器数所表示的实际数值，称为真值。

2．原码、补码和反码

原　码：一个二进制数，若用最高位表示数的符号（0 表示正数，1 表示负数），其余各位表示数值本身，则称为该二进制数的原码表示法。

补　码：一个二进制数，若以 2^n 为模，它的补码称为 2 的补码，书中 2 的补码简称补码，

即$[X]_{补}=2^n+X$。

BCD 码：二进制编码的十进制数，简称 BCD 码。

ASCII 码：ASCII 码是美国信息交换标准码，它用 7 位二进制编码来表示 128 个字符和符号。

第 2 章　微处理器结构及微型计算机工作原理

重要概念

1．微处理器、微型计算机、微型计算机系统

微处理器、微型计算机、微型计算机系统是微型计算机系统中从局部到全局的三个层次。

微处理器:简称 CPU，是用来完成运算和控制功能的部件。

微型计算机：主要由 CPU、存储器、输入输出设备及其接口电路通过总线连接而成。

微型计算机系统：以微型计算机为主体，配上系统软件和外设之后，就构成了微型计算机系统。

2．标志寄存器 FR、程序计数器 PC、堆栈指针 SP

标志寄存器 FR：寄存 ALU 操作结果的某些重要状态或特征。

程序计数器 PC：用于存放下一条要执行指令的地址码。

堆栈指针 SP：用来指示栈顶地址的寄存器。

3．堆栈

堆栈：堆栈是一种先进后出的数据结构。一般主要用于中断处理和过程调用，并有硬件堆栈与软件堆栈之分。堆栈指针 SP 用于自动管理栈区，指示当前数据存入或取出的位置。

4．逻辑地址与物理地址

逻辑地址：以段地址和段内偏移地址形式表达的某存储单元地址。表示格式为：段地址：偏移地址。

物理地址：是存储单元的实际地址编码。

说明:一个存储单元的物理地址是唯一的,而其逻辑地址不是唯一的(因段地址与偏移地址有多种组合)。

第 3 章　80x86 寻址方式与指令系统

重要概念

1．数据寻址、程序寻址

数据寻址：寻找操作数的地址。

程序寻址：寻找要执行下一条指令的地址，即程序的地址。用于程序转移或过程调用时寻找目的地址或入口地址。

2．立即数寻址和直接寻址、寄存器寻址和寄存器间接寻址的区别

立即数寻址和直接寻址的不同在于直接寻址指令中的数值是操作数的 16 位偏移地址，而不是操作数本身。为了区分二者，指令系统规定偏移地址必须用方括弧括起来。

寄存器间接寻址与寄存器寻址方式不同，指令中指定的寄存器内容不是操作数，而是操作数的偏移地址。即操作数的偏移地址放在寄存器中，操作数本身则在存储器中。用作间址的寄存器必须加上方括弧，以避免与寄存器寻址指令混淆。

3．段间转移和段内转移的区别

段间转移要改变（E）IP、CS 的值，但**段内转移**只改变（E）IP 的值，CS 的值则保持不变。

4．指令、指令系统、程序

指令：每一条基本运算或操作称为一条指令。

指令系统：一个微处理器所能执行的全部指令。指令系统的强弱是微处理器功能强弱的具体体现。

程序：按运算功能把指令排列起来，这就是程序。

5．无条件转移指令、条件转移指令、循环指令、调用指令和中断指令有什么异同？

条件转移指令是根据条件是否满足才决定是否转移。该类指令比无条件转移指令多了一个条件判断的功能，对应每种判别的条件都有一条条件转移指令。

循环指令用来管理程序循环的次数，它与条件转移指令不同的是：循环指令要对 CX 的内容进行测试，用 CX 的内容是否为 0 或把 CX 的内容是否为 0 与 ZF 标志位的状态相结合作为转移条件。

调用指令相当于先将断点地址（当前（IP）或（IP）与（CS））压入堆栈，而后再无条件转移到子程序的入口。

中断指令相当于先依次将标志位、断点地址（当前（IP）或（IP）与（CS））压入堆栈，而后再无条件转移到中断服务子程序的入口。

6．算术移位和逻辑移位指令的区别

移位指令有算术移位和逻辑移位，算术移动 N 位，相当于把二进制数乘以或除以 2^N。逻辑移位操作则用于截取字节或字中的若干位。

有符号数用算术移位，无符号数用逻辑移位。两者左移含义相同，即数值位（含符号位）依次左移，用 0 充填最低位。右移时，数值位（含符号位）依次右移，但最高位充填方法不同，算术右移用符号位充填，而逻辑右移用 0 充填。

7．用于 BCD 码和 ASCII 码算术运算的指令分别是什么？

（1）DAA（BCD 加）、DAS（BCD 减）

（2）AAA（加法的 ASCII 调整指令）

AAS（减法的 ASCII 调整指令）

AAM（乘法的 ASCII 调整指令）

AAD（除法的 ASCII 调整指令）

第 4 章　汇编语言程序设计

重要概念

1．标号与变量的区别

标号是某条指令所存放单元的符号化地址，只能在代码段中，可通过标号来引用所标识的指令（如跳转、调用指令等）；变量是数据所存放单元的符号化地址，一般位于数据段或堆栈段中。可用各种寻址方式对变量进行存取。变量和标号有段基址、偏移地址和类型三种共同属性，此外，变量还有两个特有属性：长度和字节数。

2．指令与伪指令的区别

指令语句是可执行语句，由硬件（CPU）完成其功能。伪指令是为汇编和链接程序提供编译和链接信息，属不可执行语句，其功能由相应软件完成。

3．比较宏与子程序的异同

相同点：均可用来简化源程序，并可使程序对它们多次进行调用。

不同点：

①定义方法及格式不同。
②子程序省内存，宏指令则不省。
③子程序执行速度慢，而宏运行速度快。
4．机器语言、汇编语言、高级语言、混合语言
机器语言：指令的二进制代码编程。
汇编语言：用一组字母或符号表示的指令编程。
高级语言：高级语言接近自然语言，一般总是独立于具体机器的，程序员可不必了解机器的指令系统和内部的具体结构，而把精力集中在正确掌握语言的语法规则和算法的程序实现上。一条高级语言指令对应一段汇编语言程序。
混合语言：采用两种或两种以上的编程语言加以组合编程，是一种程序接口技术，实现不同语言间的相互调用。

第 6 章　半导体存储器

重要概念
1．存储单元、存储容量、字节地址
存储单元：指存储器中每个独立地址所对应的存储空间，是计算机的基本存储器单元，一般为一个字节。
存储容量：指存储器所能容纳的最大二进制信息字节数。
字节地址：指存储器单元对应一个字节数据的地址编号。
2．寄存器、Cache、内存、外存
寄存器：寄存器包含在 CPU 内，可暂存参加运算的操作数和运算结果，以便尽可能减少 CPU 对外部存取数据的次数。其特点是 CPU 读写很快，一般在一个时钟周期中完成，寄存器数量不可能很多。
Cache：Cache 是介于内存和 CPU 之间的一种快速小容量存储器，它保留一份内存的“内容拷贝”。用来存放当前最频繁使用的程序块和数据，其特点所用存储器芯片都是高速的。作用：解决 DRAM 存储器系统的访问速度问题，有效减少 CPU 访问相对慢速的内存的次数，从而提高整机的效率。
内存：指在微型机内的存储器。存放机器正在处理及运行的指令和数据。特点速度比上两级要慢，容量比上两级要大，CPU 直接读/写。
外存：指在微型机之外、通过设备接口连接的存储器。存放当前用不着的数据、程序，用作后备存储器和作为虚拟存储器的硬件支持，一旦需要时，须先将其调入内存。其特点是容量大，但速度更慢。
　　需要说明的是，上述存储器并不是每种微机系统都具备的；另外，速度高的存储器其集成度往往低，造价高，不适用于大容量存储器中。
3．ROM、RAM
　　微机中的内存和高速缓存从功能和应用的角度分为只读（ROM）和随机读写存储器（RAM）两类。
RAM：使用时可读可写、随机存取的存储器。其特点是读写方便，使用灵活；但是一旦断电所存信息就会丢失。一般用作各种二进制信息的临时或缓冲存储。如：存放当前正在运行的程序和数据；作为 I/O 数据缓冲存储器；用作堆栈等。
ROM：使用时只读、不写的存储器。其特点是一旦写入，即使掉电，内容也不会丢失，主要用于保存存放永久性的数据。如：各种系统软件、应用程序和常数、表格等。
（1）　ROM 的分类及特点
1）分类：掩模 ROM、可编程 PROM、可擦除、可编程 EPROM、电擦除 E^2 P ROM 和闪烁存储器 FLASH。
2）特点：掩模 ROM 和 PROM 只用于大批量生产的微机产品；EPROM 和 E^2PROM 适于产品研制和小批量生产；FLASH 兼有 E^2PROM 和 SRAM 的优点，FLASH 的编程速度快，掉电后内容又不丢失。主要用作小型磁盘（如 U 盘）。
（2）　RAM 的分类及特点
1）分类：静态 SRAM、动态 DRAM 和准静态 IRAM。
2）特点：SRAM 速度高，但集成度低、成本高、功耗大，一般只用于 Cache 和小容量内存系统。DRAM

集成度高、价格低、功耗小，一般用于组成大容量内存系统。而 IRAM 兼有 SRAM 和 DRAM 的优点，应用前景较广。

4．线选法、局部译码法、全局译码法

线选法、局部译码法、全局译码法共同之处在于系统总线的低位地址线都是直接接存储器芯片的片内地址线，而其存储器片选控制则根据对余下高位地址总线的译码方案各不相同。

线选法：将余下高位地址总线分别作为各个存储器芯片的片选信号。

局部译码法：对余下高位地址总线中的一部分进行译码，译码输出作为各存储器芯片的片选控制信号。

全局译码法：对余下高位地址总线中的全部进行译码，译码输出作为各存储器芯片的片选控制信号。不同于上述两种方法的是译出地址连续，不存在地址重叠问题。

第 7 章　存储器管理

重要概念

1．实地址方式、保护虚拟地址方式、虚拟 8086 方式

实地址方式：是在加电或复位后自动建立起的一种工作方式，此方式下，80386CPU 相当于一个高速 8086/8088CPU。

保护虚拟地址方式：是一种建立在虚拟存储器和保护机制基础上的工作方式，可最大限度地发挥 CPU 性能。此方式下存储器按段组织，每段最长 4GB（2^{32} 个字节）。同时，在该方式下，80386CPU 可寻址 4GB 物理地址及 64TB（2^{46} 字节）虚拟地址空间。因而，对 64TB 虚拟存储空间允许每个任务最多可用 16K 个段。

虚拟 8086 方式：是为在保护方式下能与 8086/8088 兼容而设置的，是一种既有保护功能又能执行 8086 代码的工作方式。此方式可在实地址方式运行 8086 应用程序的同时，利用 80386CPU 的虚拟保护机构运行多用户操作系统及程序，即可同时运行多个用户程序，并能得到保护，使每个用户都感到自己拥有一台完整的计算机，非常灵活。

2．段选择子、段基地址、段描述符和段描述符表

段选择子：保护方式下，段选择器是一个指向操作系统定义的段信息的指针，可间接地提供段的基地址。

段基地址：线性空间中段的开始地址。

段描述符：描述符由 8 个字节组成，记录对段的管理信息，主要包括段基地址、段的界限、段属性。

段描述符表：存储在存储器中的数据结构。可将所有段描述符编成表。

3．80386CPU 保护模式下虚拟地址与 8086 的逻辑地址的区别

区别：

（1）段的基地址和偏移地址不再是 16 位，而是 32 位。

（2）段的基地址不直接由 16 位的段寄存器（CS，DS，ES，SS）提供，而是含在段的描述符中。每个段描述符由 8 个字节组成，其中 32 位是这个段的基地址，段的描述符存贮于称为描述符表的专门定义的存储区内。

（3）16 位的段寄存器，在保护模式下称为段选择器，其中寄存器的内容称为段的选择符。16 位段选择符中的 14 位用作索引号，依据索引号可以找到在描述符表中对应的段描述符，进而在段描述符中获得段的基地址。

4．80386 在实方式和保护方式下实现存储器寻址

（1）在实方式存储器寻址时，程序员在程序中指定逻辑地址，机器就会自动用段地址左移 4 位再加上偏移地址的方法，求得所选存储单元的物理地址，从而取得所要存储单元的内容。

（2）在保护方式存储器寻址时，程序员在程序中指定虚拟地址，机器经过分段和分页两级变换求得相应的物理地址，第一级使用段机制的描述符表，把虚拟地址转换成线性地址。第二级使用分页机制，把线性地址转换为物理地址。在这两级变换过程中，段机制是必须要用的，而分页机制则根据需要被启用

或禁止。因此，对程序员编程来说，并未增加复杂性。

5．虚拟存储器

虚拟存储器：就是系统中有一个速度较快的、容量比较小的内部主存储器，还有一个速度较慢但容量很大的外部存储器，通过存储器管理机制，使两者有机地、灵活地结合在一起，这样对于编程者来说，系统中似乎有一个容量非常大的、速度也相当快的主存储器，但它并不是真正的物理上的主存，故称为虚拟存储器。

第一章 微机基础知识

1. 进位计数制

- **表示：**

$$(N)_K = \sum_{i=-m}^{n-1} N_i \times K^i$$

其中：N_i - 第 i 位数字

n, m - n 整数位数，m 小数位数

K - 基数（采用的数字符号的个数）

K^i - 权

在汇编语言中，常用 2#,10#,16#，在数的最后用字符 B、D、H 以示区别。

- **数制转换**

10->2：整数除 2 取余,小数乘 2 取整（均从小数点开始）。 例: 115.625

10->16：10->2->16

2/16->10：按公式计算累加和（按 10 进制运算规则）

例: 11010101.011/D5.6 -> 不为 0 的位的权之和

2. 十进制数与字符的编码表示

- **BCD 码**

用 4 位 2#表示 1 位 10#,逢十进一,4 位组中各位的权=8,4,2,1

非压缩、压缩

- **ASCII 码**

3. 2#运算规则（算术、逻辑）

- **加、减、乘、除**
- **与、或、非、异或**

4. 符号数表示（重点）

- **符号表示：** 0 正，1 负，符号在最高位上
- **真值和机器数**

真值：未将符号数值化的原始数值

机器数：符号数值化的带符号数

- **机器数的表示法**

原码：符号,数值位不变。表示范围？ 0 的表示？

反码：符号,数值位变反。表示范围？0 的表示？

补码：符号,数值位变反+1。表示范围？0 的表示？

例：-123 转换成二进制数补码

• 补码的运算

利用公式：$[X+Y]_{补}=[X]_{补}+[Y]_{补}$

$[X-Y]_{补}=[X]_{补}+[-Y]_{补}$

例：X=-1000101 Y=-0110110, 求$[X+Y]_{补}$

第二章 微型计算机概述

1．微机的结构

• CPU、存储器（RAM,ROM）、I/O 接口、I/O 设备、系统总线等

· 微机与微机系统（软件的作用）

·

2． 8086/8088 微处理器

1） 最小模式下的主要引脚。功能？有效电平？是否三态？

地址线（20，8 根与数据线复用）、数据线（8）、主要控制线。

2） 内部结构。EU(通用寄存器、运算器) + BIU（段 REG、IP、地址Σ、指令队列）

3） 寄存器结构（重点）

• 通用寄存器：

数据：AX、BX、CX、DX（每个都可分为两个 8 位寄存器，共 8 个:AH，AL，...）

指针：SI、DI、SP、BP

• 控制寄存器：IP、PSW

• 段寄存器：CS、DS、SS、ES

各寄存器的主要用途？

3．存储器寻址（重点）

- 按字节（存储单元）进行存取，每次可 1/2/4 个字节
- 地址线根数与寻址范围的关系
- 存储单元内容与地址的关系
- 分段编址：

- 分段原因：要用 16 位（可寻址 2^{16}=64KB）的寄存器寻址 1MB（2^{20}）的内存空间
- 存储单元地址的 2 种表示:物理地址(唯一)和逻辑地址(不唯一)
  - 物理地址表示：20 位的真实地址
  - 逻辑地址表示：段、偏移
  - 真实物理地址的形成方法：

    段（在 CS/SS/DS/ES 中）左移 4 位+偏移地址（在 IP/SP/BP/BX/SI/SI 中）

    例:2AFBH:15ADH 所对应的真实物理地址＝？

- 段寄存器的使用
- 不同类型数据在内存中的存放顺序
- 堆栈特点及堆栈操作（PUSH、POP、CALL、RET、INT、IRET）

4. 状态寄存器 PSW(重点)

共 16 位，只用 9 位(状态 6,控制 3)

2 类:

- 状态-AF、CF、OF、PF、SF、ZF
  - 置 1/置 0 的条件?
  - 算术和逻辑操作对状态标志的影响：(算术-全部**(INC/DEC 不影响 CF)**,逻辑-PF、SF、ZF)
    - 传送操作对状态标志的影响：除 POPF/SAHF 外，均不影响标志位。
- 控制-DF、IF、TF
  - 置 1(置 0)所产生的影响?

5. 主要的 PC 总线信号

D7-D0、A19-A0、IOW#、IOR#、MEMW#、MEMR#、INTA

第三章 寻址方式和指令系统

1．寻址方式

- **什么是寻址方式：寻找、计算（取得）操作数地址的方法**
- **关于操作数的寻址方式，7 种：**

立即：操作数在指令中(不允许使用段超越)。

REG：操作数在某一寄存器中(不允许使用段超越)

*直接：操作数的地址在指令中

*REG 间接：操作数的地址在 BP/BX/SI/DI 寄存器中。串操作：源串地址固定在 SI 中、目的串地址固定在 DI 中

*寄存器相对：操作数的地址为(SI)/(DI)/(BX)/(BP)+位移量

*基+变：操作数的地址为(BX)/(BP)+(SI)/(DI)

*相对基+变：操作数的地址为(BX)/(BP)+(SI)/(DI)+位移量或

（带*的为存储器操作数）

- **说明转移地址的寻址方式，4 种：**

  段内相对：相对于 IP，地址位移量为 1 个或 2 个字节

  段内间接：寄存器或存储器的内容为转移地址，2 个字节

  段间直接：直接给出段和偏移

  段间间接：存储器的内容为转移地址，4 个字节

注意事项:

- 能够参与寻址的寄存器：BX,BP,SI,DI,DX(仅用于 I/O 指令)
- 对存储器操作数:用 BX,SI,DI 时,默认的段地址在 DS 中,用 BP 时,默认的段地址在 SS 中。若数据不在默认段中,应使用段超越。
- 指令中不可同时出现 SI/DI,也不可同时出现 BX/BP。(即一条指令中只能出现一次)

2．指令系统

指令的构成：操作码、操作数（0 个、1 个或 2 个，隐含）

• 数据传送、串操作

1)可实现 R←→R/M/Stack、Acc←→I/O、F←→Stack、Num→M/R 之间的传送。M←→M 只能用 MOVS 串指令实现。

2)包括:MOV、PUSH/POP、IN/OUT、LEA、LDS/LES 和 LODS/STOS、MOVS。

3)串操作指令的特点：

- DS:SI 寻址源,可以段超越；ES:DI 寻址目的，不允许段超越。
- 自动修改 DI / SI 的内容，增减(±)决定于 DF，增减量（1/2）由操作数的类型决定。若有串重复前缀，则也自动将 CX 减量。
- 可加重复前缀 REP，重复次数由 CX 决定。若加重复前缀 REPZ/REPNZ，则重复次数由 CX 和 ZF 共同决定。(在 LODS 指令前加串重复前缀无意义)
- 可在指令后加上 B（字节操作）或 W（字操作）。

4)段寄存器只能通过 MOV、PUSH、POP 进行操作，但修改 CS 是不合法的。

没有将立即数送入段寄存器的指令，给段寄存器赋值只能通过通用寄存器间接传送。

5)PUSH/POP 可实现 R/M/F←→S。每次传送 2 个字节。低字节在（SP-1），高字节在（SP-2）。同时使堆栈指针自动修改。

6)LEA/LDS/LES 将操作数 EA 装入寄存器/寄存器组。

7)IN/OUT 实现 Acc<->I/O,I/O 端口的地址在指令中（0-255）或在 DX 中（0-65535）。

8)凡具有两个操作数的指令——**操作数类型应相同**

•算术运算、逻辑运算和移位

1)所有这类运算的操作数都不允许是段寄存器。

2)算术/逻辑运算指令：一般要影响全部 6 个状态标志。**例外:INC/DEC 不影响 CF**。

3)ADD/ADC/SUB/SBB——8/16 位操作，不带进位/带进位。

4)INC/DEC——REG/MM（8/16 位）加/减 1。**操作数不允许是立即数**。

5)CMP 是特殊形式的减法指令。除**不存储结果**外，其他同 SUB。用于比较两个操作数之间的大小关系。

6)MUL/IMUL——8/16 位操作，结果 16/32 位。

8 位乘：操作数*AL(隐含)，结果在 AX 中

16 位乘：操作数*AX(隐含)，结果在 DX:AX 中

7)DIV/IDIV——16/32 位操作，结果 8/16 位。

8 位除：AX(隐含)/操作数→AL…AH

16 位除：DX:AX(隐含)/操作数→AX…DX

8)AAA/AAS 对不压缩的 BCD 码进行加/减法调整。**隐含操作数为 AL 和 AH**。用在加/减法指令后。

DAA/DAS 对压缩的 BCD 码进行加/减法调整。**隐含操作数为 AL**。用在加/减法指令后。

AAM 对不压缩的 BCD 码进行乘法调整。**隐含操作数为 AL 和 AH**。用在乘法指令后。

AAD 对不压缩的 BCD 码进行除法调整。**隐含操作数为 AL 和 AH**。用在除法指令前。

9)AND/OR/XOR/NOT——8/16 位。按位操作，无进借位。

10) TEST 是特殊形式的 AND 指令。除**不存储结果**外，其他同 AND。用于测试操作数的某些位的状态。(测试多个位是否为 1 的方法：先 AND，再 CMP，不用 TEST)

11) 移位和循环移位指令共有 8 种：

移位：
- 逻辑/算术左移 SHL/SAL
- 逻辑右移 SHR
- 算术右移 SAR

循环移位：
- 循环左/右移 ROL/ROR
- 带进位循环左/右移 RCL/RCR

移位位数：放在 CL 中，如：MOV CL，4；SHL AX，CL

若只移 1 位则允许以立即数形式放在指令中，如 SHL AX，1

12）串扫描 SCAS：AL/AX～ES:DI 寻址的存储单元

13）串比较 CMPS：DS:SI 寻址的存储单元～ES:DI 寻址的存储单元

14）SCAS 和 CMPS 指令前可加重复前缀 REPZ/REPNZ。重复次数由 CX 决定。REPZ 当 ZF=1 并且 CX≠0 时重复执行；REPNZ 当 ZF=0 并且 CX≠0 时重复执行。

•控制转移、处理器控制

1）标号代表了存放指令的存储单元的地址。通常作为转移指令的目标操作数。

2）无条件转移指令有 3 种：短、近、远。

短转移（SHORT）：转移范围-128～+127,段内转移

近转移（NEAR）：转移范围-32768～+32767,段内转移

远转移（FAR）：转移到系统存储器的任何位置,段间转移(跨段)

3）直接转移有 3 种：段内短/近转移，段间转移。

目的地址放在指令中，分别以运算符 SHORT、NEAR PTR、FAR PTR 予以标识。

4）间接转移有 2 种寻址方式：

目的地址放在寄存器中（近转移）：*例：JMP BX*

目的地址放在存储器中（近转移为字，远转移为双字）。

例：JMP WORD PTR[BX]； JMP DWORD PTR[BX]

5）条件转移全部是**直接短转移。**常用的有 JC/JNC、JZ/JNZ,JL/JNL,JG/JNG 等。

6）LOOP <标号>指令相当于 DEC CX/JNZ <标号>两条指令的组合。它使 CX 减 1，当 CX 不是零时转移到标号处。循环的其他形式：LOOPZ/LOOPNZ，由 CX、ZF 共同决定是否转移。

7）CALL/RET 实现过程调用和过程返回。

CALL 执行时，它把**返回地址**（即紧接在 CALL 后面那条指令的地址——IP,CS 的内容）压入堆栈，然后转移到过程。段内调用：IP 进栈；段间调用：IP、CS 都进栈。

RET 指令把返回地址从堆栈弹出到 IP（从近过程返回）或 IP、CS（从远过程返回）。

8）中断分为硬件中断和软件中断。软件中断主要是由 INT 指令产生的。用于处理中断的过程叫做中断处理程序，它是**通过中断向量间接调用**的。中断处理程序结束后，必须用 IRET 指令返回被中断的程序。

9）中断向量是中断服务程序的入口地址，4 字节（包括偏移和段地址）。

中断向量表（0-3FF，4 字节/每表项,256 个,共 1024 字节）

中断向量的偏移地址=向量号*4 (段地址＝0)

例：INT 14H, INT 2FH 的中断向量存放单元=?

10）CPU 响应中断后，将 PSW、IP 和 CS 压入堆栈，清除 T 和 I 标志位，然后根据中断类型号从中断向量表取出中断向量送到 CS、IP，从而转到相应的中断处理程序执行。

11）中断处理结束，用 IRET 指令从堆栈中恢复返回地址（IP，CS）和标志寄存器 PSW。

12）中断允许标志位（IF）控制 CPU 的 INTR（可屏蔽中断请求）引脚。STI 允许 CPU 响应可屏蔽中断请求，CLI 则禁止。IF 标志位不影响软中断（INT 指令）。

13）CLC/STC/CMC 用于清除/置位/取反 CF 标志。

14）CLD/STD 指令清除/置位 DF 标志。它控制串操作的方向。DF=0 地址增量，DF=1 地址减量。

第四章　汇编语言程序设计

- **顺序程序设计**

要注意语法、汇编语言框架、算法逻辑等方面符合要求。

- **分支、循环程序设计**

转移条件（CF、ZF、OF、SF、PF)、指令用法（根据标志位）

相关指令：Jx；x=C/NC(B/NB)、Z/NZ、G/NG、L/NL、GE/NGE、LE/NLE、A/NA、O/NO、P/PO、S/NS

LOOP

REP/REPZ/REPNZ

3）**子程序设计（重点是参数传递方法和堆栈概念）**

子程序结构（框架)：PROC … ENDP

相关指令：CALL、RET

保护/恢复寄存器：入口处和退出前，PUSH/POP

汇编源程序

- **语句格式：[标号:]　操作码　操作数　；注释**
- **各种名字的定义规则（包括段名、标号、变量名、常量名、过程名)：**

（1）　以包括 A-Z、0-9 和“？.@_$ ”5 个特殊字符

（2）　不能以数字开头

（3）　不能与保留字（指令助记符/伪指令/寄存器名等）重名，使用标号时特别要注意

（4） 不能重复定义

（5） 不能超过 31 个字符

- **汇编语言程序的结构（框架）**

```
DATA    SEGMENT
        <数据定义伪操作>
DATA ENDS
CODE    SEGMENT
        ASSUME CS:CODE,DS:DATA
START:  MOV  AX,DATA
        MOV  DS,AX
         . . .
        MOV  AH,4CH
        INT  21H

CODE    ENDS
        END START
```

注意：主程序开始处要设置 DS、ES 段寄存器

- **数据项和表达式**

操作数构成：REG、存储器单元、数据项

数据项：常量、标号、变量以及三者的组合(表达式)

- 常量：数字常量 2#、16#、10#，字符常量‘XXX’
- 标号：某条指令的符号地址,定义时要加冒号（:），引用时不要冒号
- 变量：内存中的数据区

  定义格式：[变量名] DB(DW,DD) [表达式]

- 表达式:算术、逻辑、关系

  算术：+、-、*、/

  逻辑：AND、OR、NOT、XOR

  关系：EQ、NE、LT、GT、LE、GE（结果是逻辑值，真-全 1，假-全 0）

```
例: MOV  AX,4*64          ;AX←256
    MOV  AX,40H EQ 64 ;AX←0FFFFH
```

- 类型操作符：（BYTE、WORD、DWORD、NEAR、FAR）PTR
- 取段、偏移地址操作符：SEG、OFFSET（变量和标号）

```
例: MOV  BX,OFFSET VAR ;BX←VAR 的偏移地址(或 EA)
    MOV  AX,SEG VAR    ;AX←VAR 的段地址
```

（注意与 LEA 指令的异同）

- **指示性语句**

变量定义：DB、DW、DD、DUP

符号赋值：EQU、=

段定义：SEGMENT/ENDS、ASSUME

定位类型：BYTE、WORD、PARA、PAGE

组合类型：PUBLIC、AT、STACK

类别名：‘CLASS’（段组名）

伪指令 SEGMENT 指定一个段的开始,ENDS 指定一个段的结束。SEGMENT/ENDS 必须成对使用。

伪指令 ASSUME 通知汇编程序 CS、DS、ES、SS 与逻辑段之间的关系——用于语法检查。

- **DOS 功能调用（INT 21H,AH=功能号，参数传递：寄存器）**
- **方法：**

（1） MOV AH，功能号
（2） 设置入口参数
（3） INT n
（4） 分析出口参数

- **举例：**

1） 输入一个字符：01H，AL=字符

2）输入字符串：0AH，DX=缓冲区地址(格式

Byte 1	Byte 2	字节数=缓冲区大小
缓冲区大小	实际键入字符数	输入缓冲区

)

```
例: BUFSIZE DB  n          ;n=<缓冲区大小>
    ACTSIZE DB  ?          ;实际键入字符数将访在此单元
    BUFFER  DB  n DUP(?)   ;预留 n 个字节
```

3）显示一个字符：02H，DL=字符

4）显示一个字串：09H，DX=字串首址(字串以'$'结束)

5）返回 DOS 系统：4CH

- **宏指令:**定义和使用

  定义： **宏定义名 MACRO　　[形参表]**

  <宏定义体>

  ENDM

  引用： **宏定义名　[实参表]**

可以把“宏”当作指令一样使用,所以也称其为**“宏指令”**。

第六章 半导体存储器

1. 要求

1) 分类、特点、工作原理——**ROM**（EPROM，EEPROM），**RAM**(SRAM，DRAM)，**CACHE**

2) 计算构成存储器所需的芯片的数量，位扩、字扩的连接方法

3) 利用全译码/部分译码，将存储器芯片连接到系统总线上并映射到任意地址空间

4) 给出存储器的地址范围求内存的首址/末址和容量，反之也要会求解

5) 根据地址范围用门电路（包括 74LS138）构成译码器的方法

2. 重点

1) 基本概念

- 存储器存储数据的单位：

bit——存储数据的**最小单位**，能存储 1 位二进制代码（0/1），简写为 b

Byte——存取数据的**基本单位**，也称存储单元，能存储 8 位二进制代码（0/1），简写为 B

内存以字节为单位进行存取，字长为 16、32、64 位时，则每次访问 2/4/8 个单元

- 存储器的容量：微机系统中存储单元的总和。单位为 KB、MB、GB
- 存储器的编址：地址——给每个存储单元分配的顺序编号
- 寻址范围与地址总线位数的关系：地址线 n 根，地址范围 2^n，分布在 0～2^n-1
- 寻址范围与存储容量的关系：寻址范围——CPU 编址单元的总和，定值

  存储容量——实际配置的物理内存单元的个数，可变

2) 存储芯片的容量：

用“单元数×每单元位数”表示

3) 字扩、位扩及连接方法

位扩——对位数扩充，如用存储芯片制造内存条

字扩——对容量扩充，如往微机中扩充内存条

字位扩——对位数、容量均扩充

连接规则：位扩——数据线分别引出，地址/控制线并联

字扩——芯片地址线/读写信号/数据线并联，片选信号接高位地址的译码输出

4） 地址译码

存储芯片的容量总是小于 CPU 的寻址空间。也就是说，一个芯片的地址线数总是少于系统地址总线的根数。

系统地址总线分为两部分：低位地址线直接接存储芯片；高位地址线则用于片选译码（称为地址译码），用于选择哪个存储芯片工作。

全地址译码——全部高位地址线都参加片选译码

部分地址译码——部分高位地址线参加片选译码（一个芯片会对应多个地址范围）

常用译码电路——用与、或、非门构建，或用 74LS138 等现成的译码器

（牢记 74LS138 的引脚及使用方法）

5） 地址范围、内存容量、首址/末址

利用公式： 末址 = 首址＋容量(字节数)－1

容量 = 末址－首址＋1

再记住 3 个关键数字：0～000FFH → 00100H = 256B

0～00FFFH → 01000H = 4KB

0～0FFFFH → 10000H = 64KB

例：A8000H～CFFFFH， 容量 = CFFFFH-A8000H+1 = 28000H = 64K×2+4K×8 = 160K

容量 64KB， 首址 BC000H， 末址 = BC000H＋10000H-1 = CBFFFH

6） 存储芯片与总线的连接

重点是 SRAM 和 EPROM 与系统总线的连接方法。

数据线、地址线(包括译码器)、CS#/CE#、OE＃、(BUSY/READY)

第八章 中断

1. 中断

1）基本概念

- 中断、中断类型、中断向量的作用
- CPU 的有关引脚——INTR、NMI（区别？满足什麽条件 CPU 才能响应）
- 中断处理的基本过程

请求——→判优——→响应——→处理——→返回

8259：判优、排队、送中断向量（类型）码
CPU：响应请求、根据向量码获得中断向量

CPU：执行中断服务程序，
执行完后返回断点

2）8086/8088 中断系统——

- 根据中断类型码（向量码）获得中断向量的方法
- 中断向量表的位置与结构

3）8259

- 8259 的初始化——ICW1-4、OCW1-3 的作用，初始化顺序
- 8259 的寻址和连接——各寄存器如何寻址，N 片可接 7*N+1 中断

第九章 输入输出方法及常用的 I/O 接口芯片

1.概述

1）I/O 端口的编址

- 端口地址：I/O 接口中的子部件或寄存器的编号
- 两种编址：统一编址、独立编址，特点及优缺点
- 8088/8086 系统的 I/O 寻址:直接寻址 256 个(0～0FFH)，间接寻址（DX）64K 个（PC 中只允许 1024 个,0～3FFH）

2）输入输出的基本方法

无条件、**查询**、中断、DMA

- 查询——用 I/O 指令读入外设状态（数据准备好/设备空闲），以决定能否进行读写
- 中断——外设准备好时主动向 CPU 提出申请，CPU 暂停当前程序，转入中断服务程序，完成数据传送。中断服务完毕，CPU 返回被中断的程序继续执行
- DMA——CPU 暂时把总线控制权交由 DMAC 掌管，数据传送完后再把总线控制权交还给 CPU

重点：查询方式的工作流程（读状态端口、判断准备好否、读/写数据端口），**编程方法**？

3）系统总线中与输入输出有关的主要信号线

D0-D7、A15-A0、IOW#、IOR＃、INTR、INTA

2.简单接口

- 三态门、锁存器，适用场合（输入/输出，简单外设）
- 外设状态的测试方法，如何编程

常用接口芯片：

1）8255

- 工作方式：重点是**方式 0、1**，方式 2 仅要求了解

- 寻址——内部 3 个通道和控制寄存器如何寻址
- 引脚连接：面向系统总线的引脚、面向 I/O 设备的引脚（PA/PB/PC 三组）
- 初始化——写控制字（各位的含义）、C 口的位控字用法

注意：C 口分为两个 4 位组，同一组内只能**都是输入**或**都是输出**。C 口通常用作控制信号的输出和状态信号的输入。用作输出时，C 口既可以用写位控制字的方法输出，也可以用标准的往 C 口写数据的方法输出。

2）8253

- 注意 CLK、OUT、GATE 和计数初值四者的关系。
- 从 6 个方面进行总结：**启动方法、自动重复否、写控制字后 OUT 的变化、GATE 作用、正在计数时写入新的初值何时生效、OUT 输出波形**
- 工作方式——0～5，重点是方式 0－3，参考“工作方式一览表”

  0：产生宽度约为$(N+1)*T_{CLK}$的负脉冲，软件启动，不自动重复

  1：产生宽度约为$N*T_{CLK}$的负脉冲，硬件启动，不自动重复

  2：产生周期为$N*T_{CLK}$，宽度为T_{CLK}的负脉冲，硬件/软件启动，自动重复

  3：产生周期为$N*T_{CLK}$的方波，硬件/软件启动，自动重复
- 寻址和连接——内部 3 个计数器和控制寄存器如何寻址，与总线如何连接，如何控制 GATE
- 初始化——写控制字（各位的含义）、写计数初值（8 位/16 位），每个计数器分别初始化

要求重点掌握 8253 和 8255 的控制字，理解各位的意义

3）打印机查询

- 打印机
  - 工作时序

    打印机—启动方法？如何知道打印完成？怎样打印结果？
  - 引脚与联接方法（**重点：如何用 8255 作为接口进行控制**）

    重点：如何连接？怎样编程进行控制？