汇编

1. 题型
   - 选择 10分
   - 填空 15分
   - 判断 5道10分
   - 简答 5道25分
   - 程序理解 4道20分
   - 编程题 2道20分
2. 可能会考的题目
   - 汇编语言的优缺点
     - 优点：利用汇编语言可能编写出在“时空”两个方面最有效率的程序。另外，通过它可最直接和最有效地操纵机器硬件系统 。
     - 缺点：它面向机器，与机器关系密切，它要求程序员比较熟悉机器硬件系统，要考虑许多细节问题。最终导致程序员编写程序繁琐，调试程序困难，维护、交流和移植程序更困难。
   - A a Z z的ASCII码十进制、十六进制
     - 十进制数的编码就是对应的十进制数字加上30H，对于大小写字母的编码相差20H
     - A:41H, 65 a:61H, 97 Z:5AH, 90 z:7A, 122
   - 标志寄存器中有哪些状态标志？这些状态标志的主要作用是什么？
     1. CF(进位标志carry flag): 对无符号数比较重要，如果运算结果的最高位产生一个进位或借位，则CF被置1，否则CF被清0.
     2. ZF(零标志zero flag)：如果运算结果为0，则ZF被置1，否则ZF被置0
     3. SF(符号标志sign flag)：如果运算结果的最高位为1，则SF被置1，，否则SF被清0
     4. OF(溢出标志overflow flag)：对有符号数比较重要，如果运算结果超出有符号数的表示范围，OF被置1，否则OF被清0
     5. PF(奇偶标志parity flag)：反映运算结果的最低字节中含有“1”的位数是偶数还是奇数。如果“1”的位数是偶数则置1，否则置0
     6. AF(辅助进位标志)：如果产生进位或者借位，则AF被置1，否则AF被清0
   - 哪些汇编指令不改变标志寄存器
     - mov
     - lea
     - 条件转移指令
   - 寻址方式
     1. 立即寻址方式：mov eax，1
     2. 寄存器寻址方式：mov eax，ebx
     3. 直接寻址方式：mov eax， [12340H]
     4. 寄存器间接寻址方式：mov eax，[ebx]
     5. 基址+n*变址+偏移量
        - 寄存器相对寻址：mov eax， [ebx+4]
        - 基址加变址寻址：mov eax，[ebx+ecx]
        - mov eax, [ebx+2*ecx+4]
   - 显式指定存储器操作数的尺寸
     - mov word ptr [ebx], 4

    - mov dword ptr [ebx], 4
    - mov byte ptr [ebx], 4
- 如何比较数值大小
    - 如果两者是无符号数则可以根据进位标志cf判断大小
    - 如果两者是有符号数则同时根据符号标志sf和溢出标志of判断大小
    - 有符号数：G E L，
    - 无符号数：A E B，
    -
- 转移指令的作用
- 转移指令可以分为哪几类（条件转移指令与循环指令仅仅限于段内转移）
    1. 段内转移：仅仅设置EIP
        - 段内直接转移
        - 段内间接转移
    2. 段间转移：设置了EIP和CS
        - 段间直接转移
        - 段间间接转移
- 堆栈的4个作用并**举出例子**
    - 保存函数的返回地址
    - 向函数传递参数
    - 安排函数局部变量
    - 保护现场
- 为什么寄存器作为局部变量可以提高效率
    - 寄存器在CPU内部，处理寄存器中的数据要比处理存储器中的数据快的多
- 寄存器、堆栈传参的优缺点是什么
- CPU与外设之间交换信息包括哪三类
- 谈谈你对汇编语言的看法

3. 4.2、4.3、第五章、7.2、7.3、7.4不考
4. jmp与call的区别
    - call会将返回地址压入堆栈
5. 一些指令
    - mov：两个操作数不能同时为存储单元(8、16、32)
    - xchg：可以为通用寄存器或者存储单元，但不能同时为存储单元(8、16、32)
    - add：与mov相同
    - sub：与add相同
    - inc：通用寄存器或者存储单元
    - dec
    - neg：取操作数的负数
    - clc stc cmc
    - LAHF：把标志寄存器的低8位送到通用寄存器AH中，漏调了OF
    - SAHF
    - adc：des=des+src+cf
    - sbb

- lea：源操作数必须是一个存储器操作数， 目的操作数必须是16位或者32位的通用寄存器
- cmp
- **push：操作数为32位通用寄存器、16位通用寄存器、段寄存器、双字存储单元、字存储单元、立即数**
- **pop：没有立即数**
- pusha popa pushad popad
- mul ：无符号乘法指令，操作数**不能是立即数**，隐含的操作数在al ax eax中，结果分别送到ax dx：ax edx：eax中
- imul：有符号乘法指令
  - imul oprd：与mul类似
  - imul dest, src
  - imul dest, src1, src2
- div：无符号除法指令，隐含的操作数在ax dx：ax edx：eax中，商送到al ax eax，余数送到ah dx edx中
- idiv：有符号除法指令，与div相同
- 符号扩展指令
  - cbw: al===>ax
  - cwd: ax===>dx:ax
  - cdq: eax===>edx:eax
  - cwde
- 扩展传送指令
  - movsx dest， src：把src符号扩展后送至目的操作数dest，目的操作数只能为通用寄存器且必须比src位数大
  - movzx dest，src：把src零扩展后送至目的操作书dest
- 逻辑运算指令：not and or xor test
- 移位指令： 指令 OPRD， count(8位立即数或者存储器cl)
  1. sal shl：算术左移 逻辑左移相同
  2. sar：算术右移
  3. shr：逻辑右移
  4. 循环移位
     - rol
     - ror
     - rcl
     - rcr
- loop
- loope/loopz(ecx!=0 and zf = 1)
- loopne/loopnz(ecx!=0 and zf = 0)
- jecxz
- 字符串操作指令
  1. lodsb lodsw lodsd esi ===> al || ax || eax
  2. stosb stosw stosd al || ax || eax ===> edi
  3. movsb movsw movsd
  4. scasb scasw scasd al与edi
  5. cmpsb cmp sw cmpsd esi与edi

6. rep重复前缀 每一次都判断ecx是否等于0

7. repe repz repne repnz

8. df = 0由低到高 df = 1由高到低 cld std