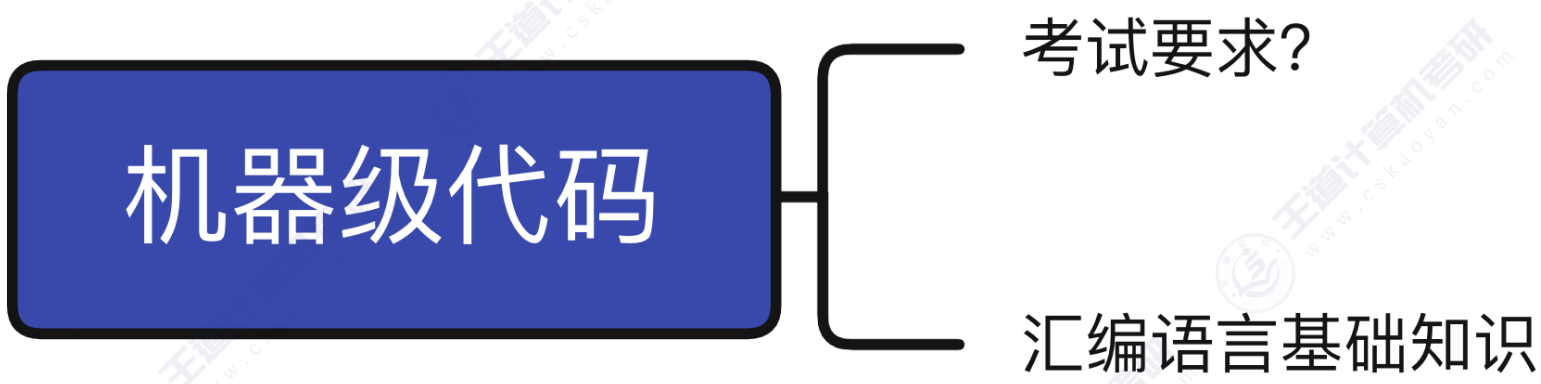
本节内容

高级语言与机器级代码之间的对应

本节总览



高级语言→汇编语言→机器语言

考试要求：

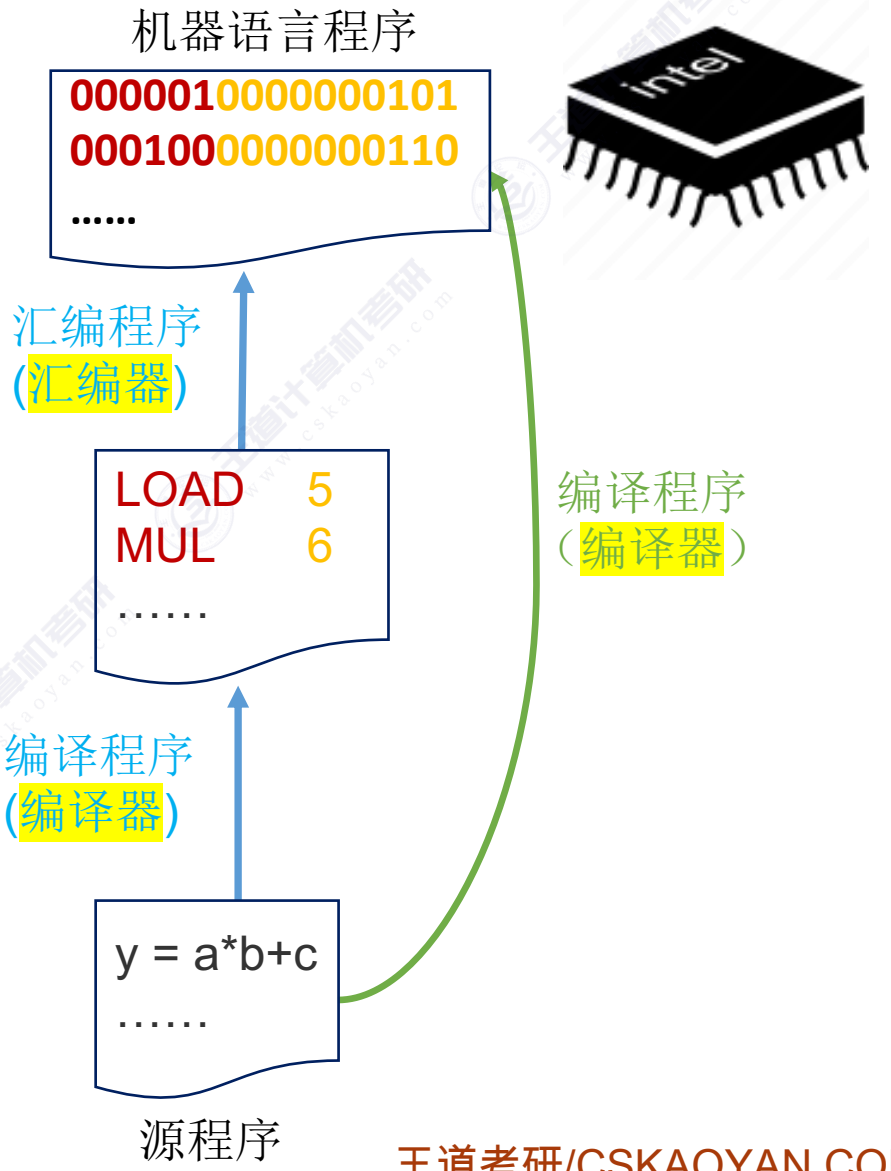
- 只需关注x86汇编语言；若考察其他汇编语言题目会详细注释
- 题目给出某段简单程序的C语言、汇编语言、机器语言表示。能结合C语言看懂汇编语言的关键语句（看懂常见指令、选择结构、循环结构、函数调用）
- 汇编语言、机器语言 一一对应，要能结合汇编语言分析机器语言指令的格式、寻址方式
- 不会考：将C语言人工翻译为汇编语言或机器语言

机器级代码

机器语言：二进制代码

汇编语言：助记符

高级语言：C



历年真题

2.【2019 统考真题】已知 $f(n)=n!=n\times(n-1)\times(n-2)\times\cdots\times2\times1$，计算 $f(n)$的 C 语言函数 f1 的源程序（阴影部分）及其在 32 位计算机 M 上的部分机器级代码如下：

```
int   f1(int n){
  1     00401000    55              push  ebp
        ...         ...             ...
        if(n>1)
  11    00401018    83 7D 08 01     cmp  dword ptr [ebp+8],1
  12    0040101C    7E 17           jle  f1+35h (00401035)
        return n*f1(n-1);
  13    0040101E    8B 45 08        mov  eax, dword ptr [ebp+8]
  14    00401021    83 E8 01        sub  eax, 1
  15    00401024    50              push  eax
  16    00401025    E8 D6 FF FF FF  call  f1 ( 00401000)
        ...         ...             ...
  19    00401030    0F AF C1        imul  eax, ecx
  20    00401033    EB 05           jmp  f1+3Ah (0040103a)
        else return 1;
  21    00401035    B8 01 00 00 00  mov  eax,1
}
        ...         ...             ...
  26    00401040    3B EC           cmp  ebp, esp
        ...         ...             ...
  30    0040104A    C3              ret
```

其中，机器级代码行包括行号、虚拟地址、机器指令和汇编指令，计算机 M 按字节编址，int 型数据占 32 位。请回答下列问题：

x86汇编语言指令基础

Intel x86架构CPU

起源：代号为 8086 的CPU

指令的作用？

- 改变程序执行流
- 处理数据

指令格式：操作码 + 地址码

- 怎么处理？
- 数据在哪儿？
  - 在寄存器里 → 在指令中给出“寄存器名”
x86架构的CPU有哪些寄存器？
  - 在主存里 → 在指令中给出“主存地址”
怎么在指令中指明读写长度？
  - 在指令里 → 直接在指令中给出要操作的数
也就是“立即寻址”

以 mov 指令为例

destination：目的地

source：来源、发源地

mov 目的操作数d, 源操作数s

#mov指令功能：将源操作数s复制到目的操作数d所指的位置

mov eax, ebx　　#将寄存器 ebx 的值复制到寄存器 eax

mov eax, 5　　#将立即数 5 复制到寄存器 eax

mov eax, dword ptr [af996h]　　#将内存地址 af996h 所指的32bit值复制到寄存器 eax

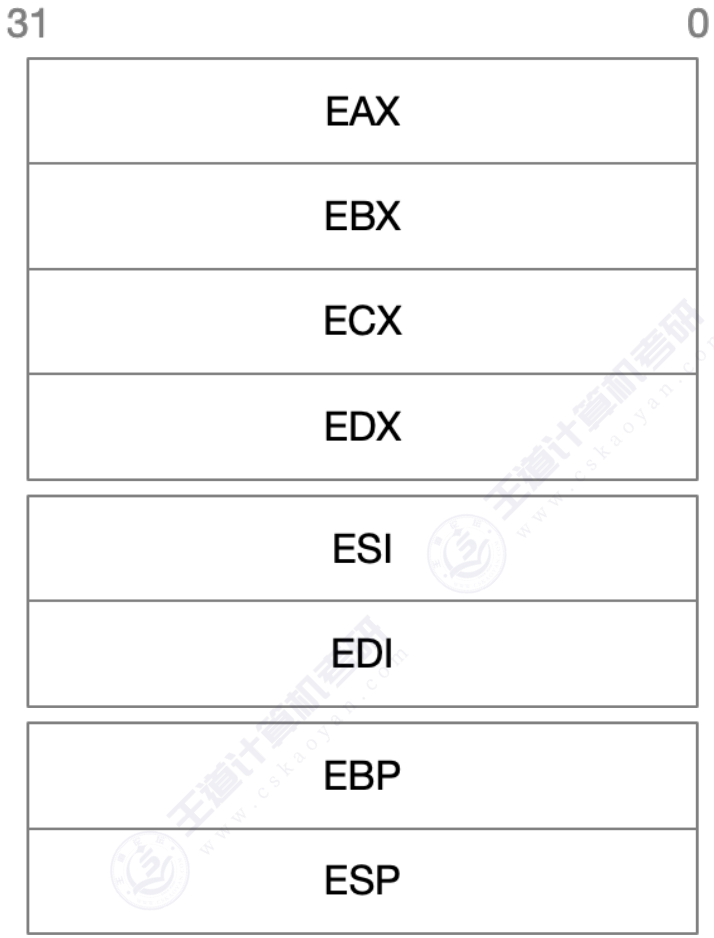
mov byte ptr [af996h], 5　　#将立即数 5 复制到内存地址 af996h 所指的一字节中

如何指明内存的读写长度：

dword ptr——双字，32bit
word ptr——单字，16bit
byte ptr——字节，8bit

x86架构CPU，有哪些寄存器？

每个寄存器都是32bit



通用寄存器（X = 未知）
E = Extended = 32bit

变址寄存器（I = Index）
S = Source，D=Destination

堆栈基指针（Base Pointer）

堆栈顶指针（Stack Pointer）

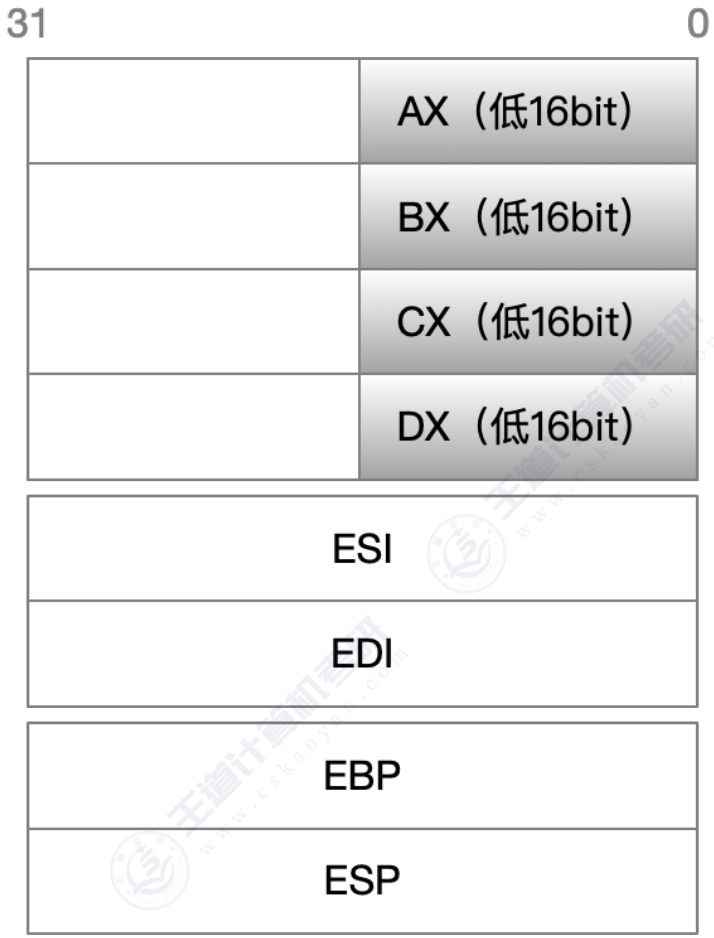
```
mov eax, ebx                    #寄存器→寄存器
mov eax, dword ptr [af996h]     #主存→寄存器
mov eax, 5                      #立即数→寄存器
```

变址寄存器可用于线性表、字符串的处理

堆栈寄存器用于实现函数调用

x86架构CPU，有哪些寄存器？

每个寄存器都是32bit



通用寄存器，使用低16bit

```
mov ax, bx                          #寄存器→寄存器
mov ax, word ptr [af996h]           #主存→寄存器
mov ax, 5                           #立即数→寄存器
```

变址寄存器（I = Index）
S = Source，D=Destination

两个变址寄存器只能固定使用32bit

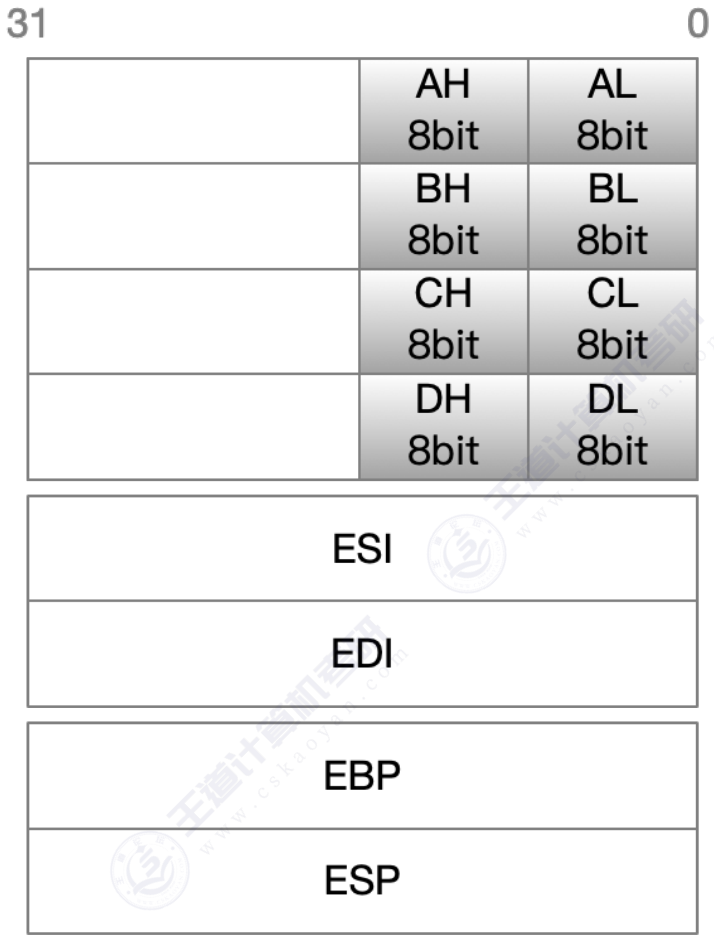
堆栈基指针（Base Pointer）

堆栈顶指针（Stack Pointer）

两个堆栈寄存器只能固定使用32bit

x86架构CPU，有哪些寄存器？

每个寄存器都是32bit



通用寄存器，使用指定8bit

```
mov ah, bl                      #寄存器→寄存器
mov ah, byte ptr [af996h]       #主存→寄存器
mov ah, 5                       #立即数→寄存器
```

变址寄存器（I = Index）
S = Source，D=Destination

两个变址寄存器只能固定使用32bit

堆栈基指针（Base Pointer）

堆栈顶指针（Stack Pointer）

两个堆栈寄存器只能固定使用32bit

更多例子

```
mov eax, dword ptr [ebx]          #将 ebx 所指主存地址的 32bit 复制到 eax 寄存器中
mov dword ptr [ebx], eax          #将 eax 的内容复制到 ebx 所指主存地址的 32bit
mov eax, byte ptr [ebx]           #将 ebx 所指的主存地址的 8bit 复制到 eax
mov eax, [ebx]                    #若未指明主存读写长度，默认 32 bit
mov [af996h], eax                 #将 eax 的内容复制到 af996h 所指的地址（未指明长度默认32bit）

mov eax, dword ptr [ebx+8]        #将 ebx+8 所指主存地址的 32bit 复制到 eax 寄存器中
mov eax, dword ptr [af996-12h]    #将 af996-12 所指主存地址的 32bit 复制到 eax 寄存器中
```

总结

指令的作用？
- 改变程序执行流
- 处理数据

指令格式：操作码 + 地址码
- 怎么处理？
- 数据在哪儿？
  - 在寄存器里
    - 在汇编指令中，给出寄存器名
    - 通用寄存器：eax、ebx、ecx、edx
    - 变址寄存器：esi、edi
    - 堆栈寄存器：ebp、esp
  - 在主存里
    - 在汇编指令中，给出读写长度、主存地址
    - dword ptr [地址] #32bit
    - word ptr [地址] #16bit
    - byte ptr [地址] #8bit
  - 在指令里
    - 在汇编指令中，直接给出常量，即“立即寻址”
    - 可用十进制表示、也可用十六进制（常以h结尾）

公众号：王道在线

b站：王道计算机教育

抖音：王道计算机考研