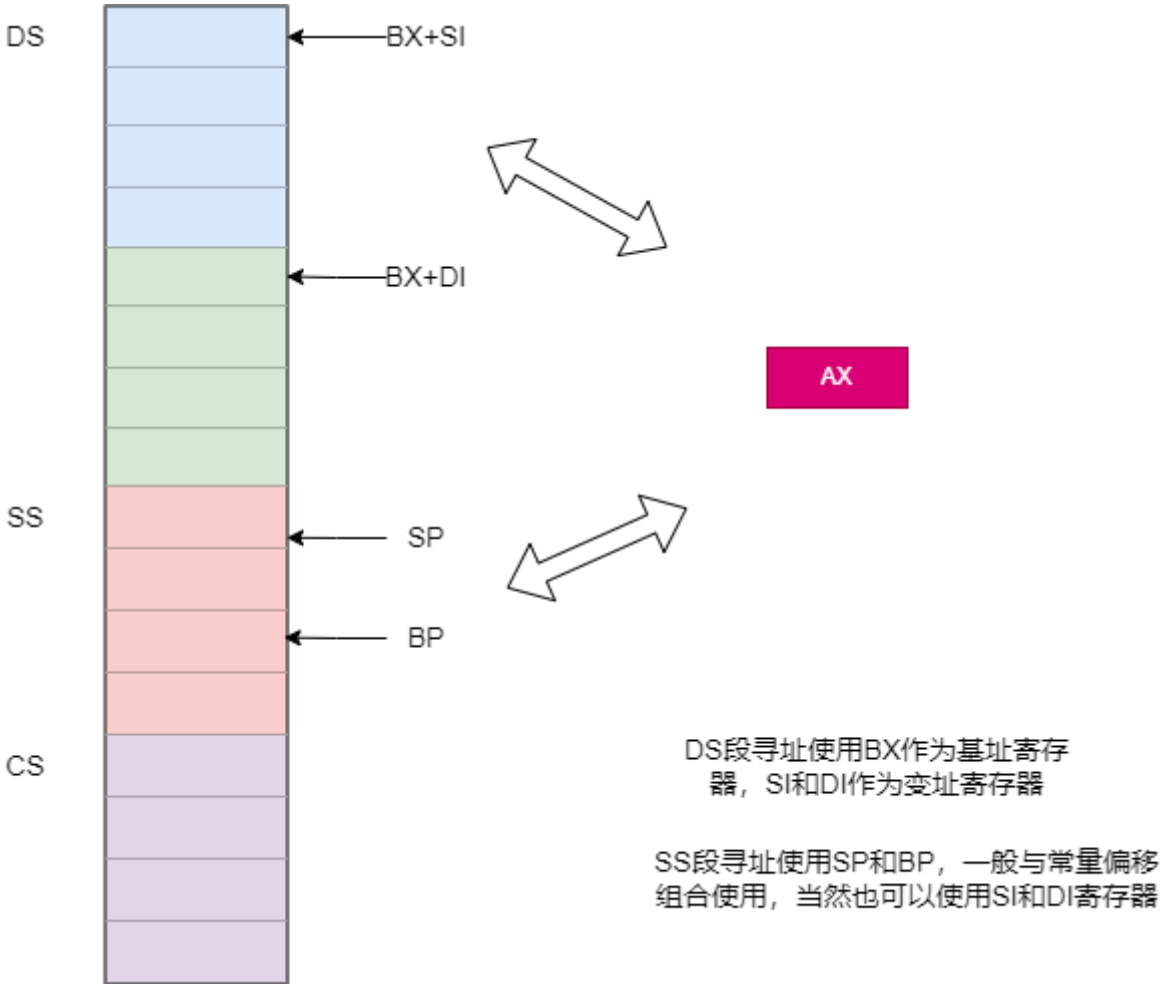
第1节 汇编程序入门

写在前面：写汇编程序，脑子里一定要有这样一张图



1.程序结构

example01.asm

```
assume cs:code,ds:data,ss:stack,es:edata

data segment
    str1: db 'hello,world!'   ;需要复制的源字符串
data ends

edata segment
    str2: db 12 dup('?')      ;需要复制到的地方
edata ends

code segment
main:
    mov ax,data               ;初始化数据段寄存器
    mov ds,ax

    mov ax,edata              ;初始化附加段寄存器
    mov es,ax
```

```
    mov si,offset str1      ;初始化源指针
    mov di,offset str2      ;初始化目标指针

    mov cx,6                ;需要执行循环6次

copy:                        ;循环的起始地址
    mov ax,[si]
    mov es:[di],ax
    add si,2
    add di,2

    loop copy               ;开始循环
end main
code ends
```

指定段名

```
assume cs:code,ds:data,ss:stack,es:edata
```

编写每段的代码

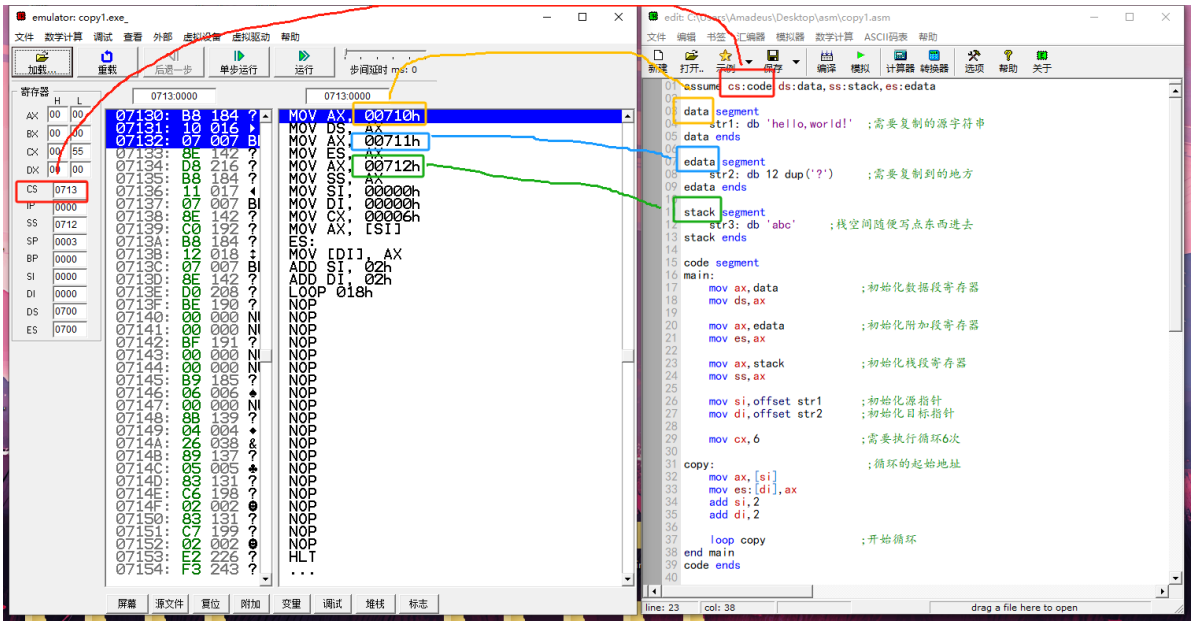
```
data segment
    ; data段的代码
data ends
```

```
code segment
    ; code段的代码
code ends
```

然后对程序进行编译，汇编程序会将每段的内容进行处理，然后写入内存相应的位置，最后执行代码

2.汇编程序处理代码的大致过程

第一步，指定各个段的地址



```
assume cs:code,ds:data,ss:stack,es:edata
```

上面这个是程序编译完成之后，程序执行之前，各个段寄存器的状态，编译器首先给定义的4个段分别分配一个地址，

```
code:0713h
data:0710h
edata:0711h
stack:0712h
```

然后会使用code段的地址初始化cs段寄存器

```
cs：0713h
```

其他的段寄存器随便初始化，因此程序开头需要手动把各个段寄存器用段名初始化一下，保证后面使用方便；

通用寄存器默认初始化为0；

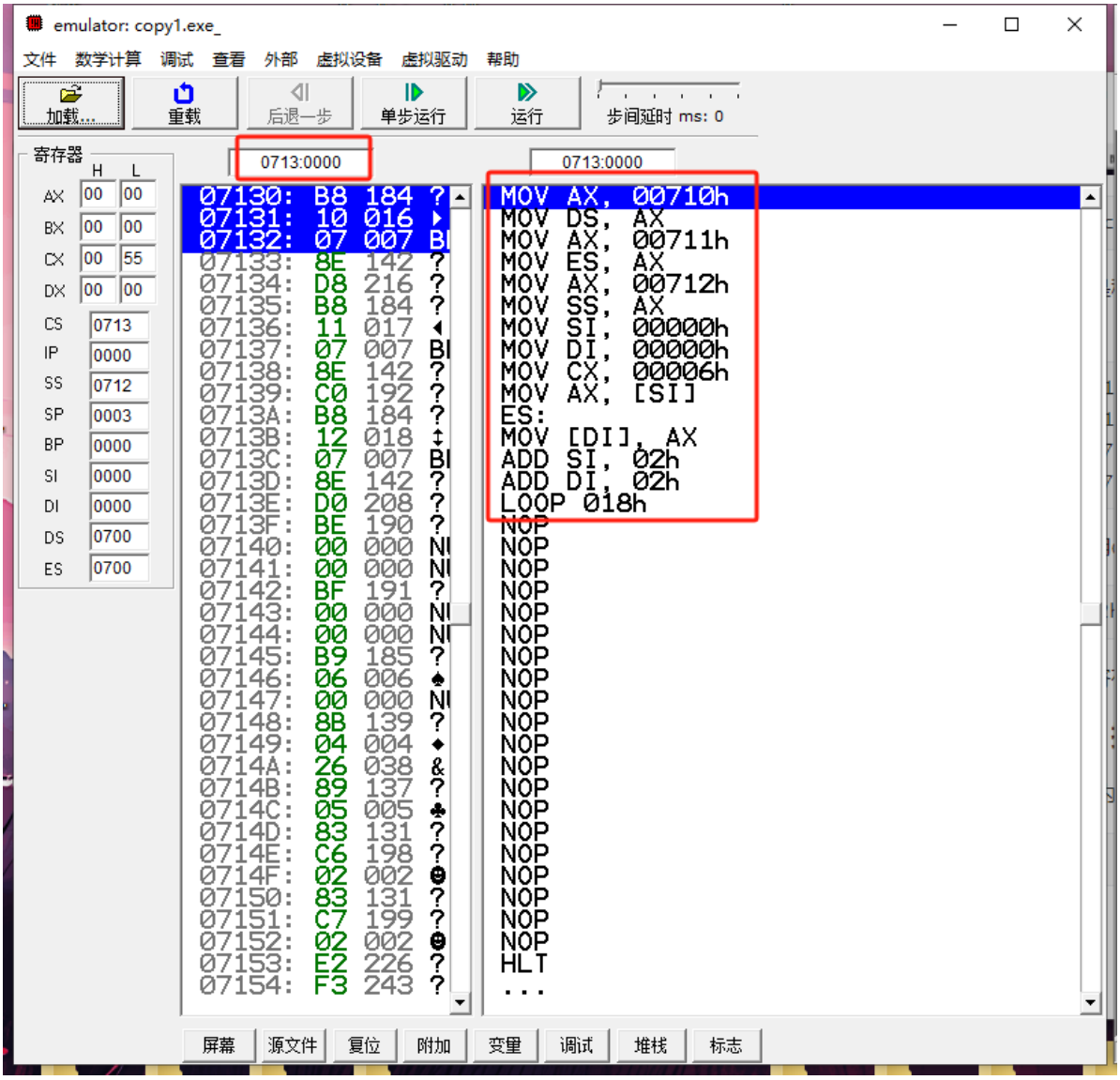
指令寄存器IP，是程序执行的入口地址，默认初始化为0，但是我们这里在代码段中指定了main函数，

并且以end main结尾，会让main的偏移地址来初始化IP寄存器，由于code段第一句代码就是main开头的，

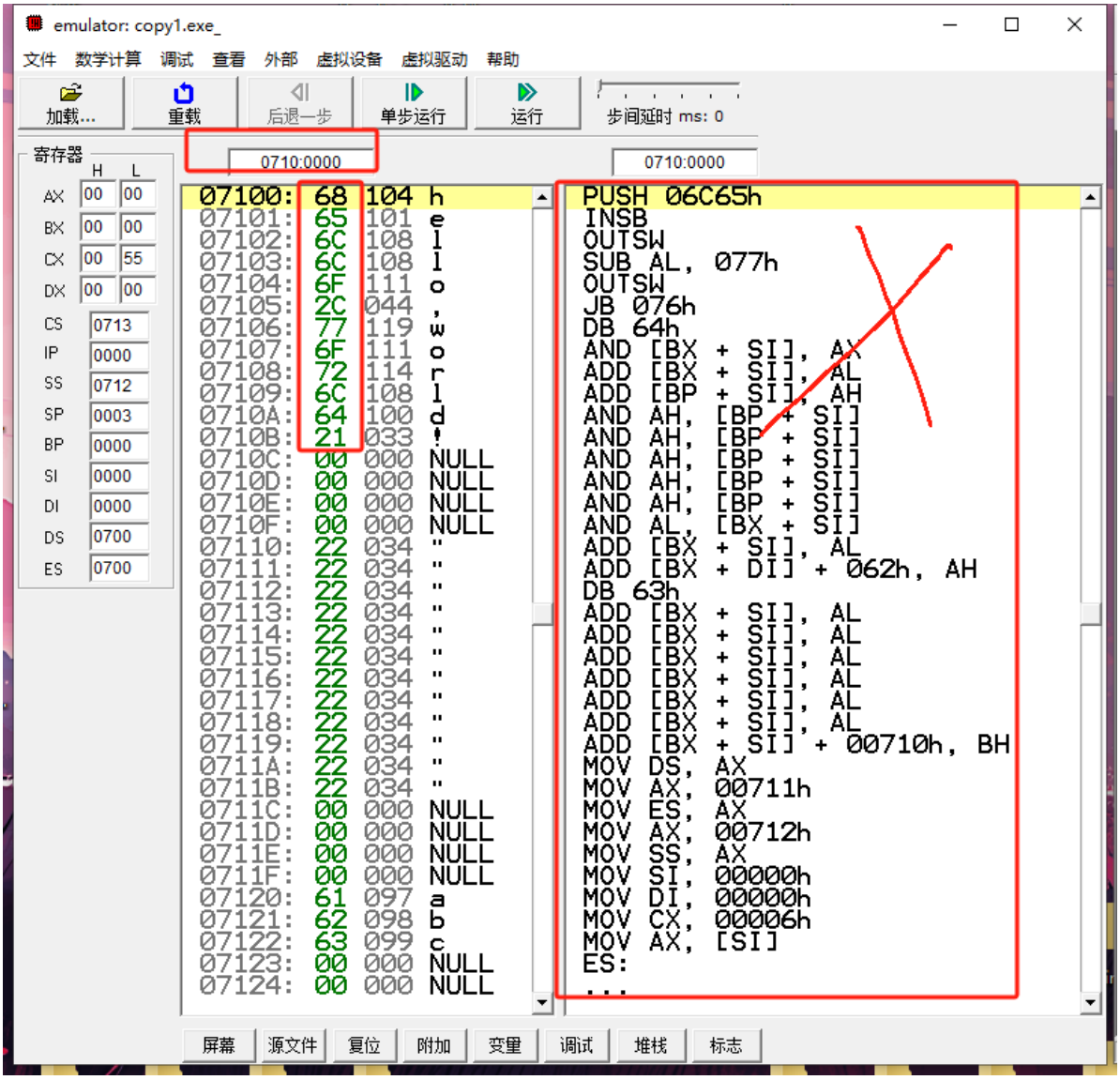
所以IP寄存器还是为0。

第二步，把每个段的内容写入相应的地址

code段写入的内容如下(也就是0713h开头的地址)：



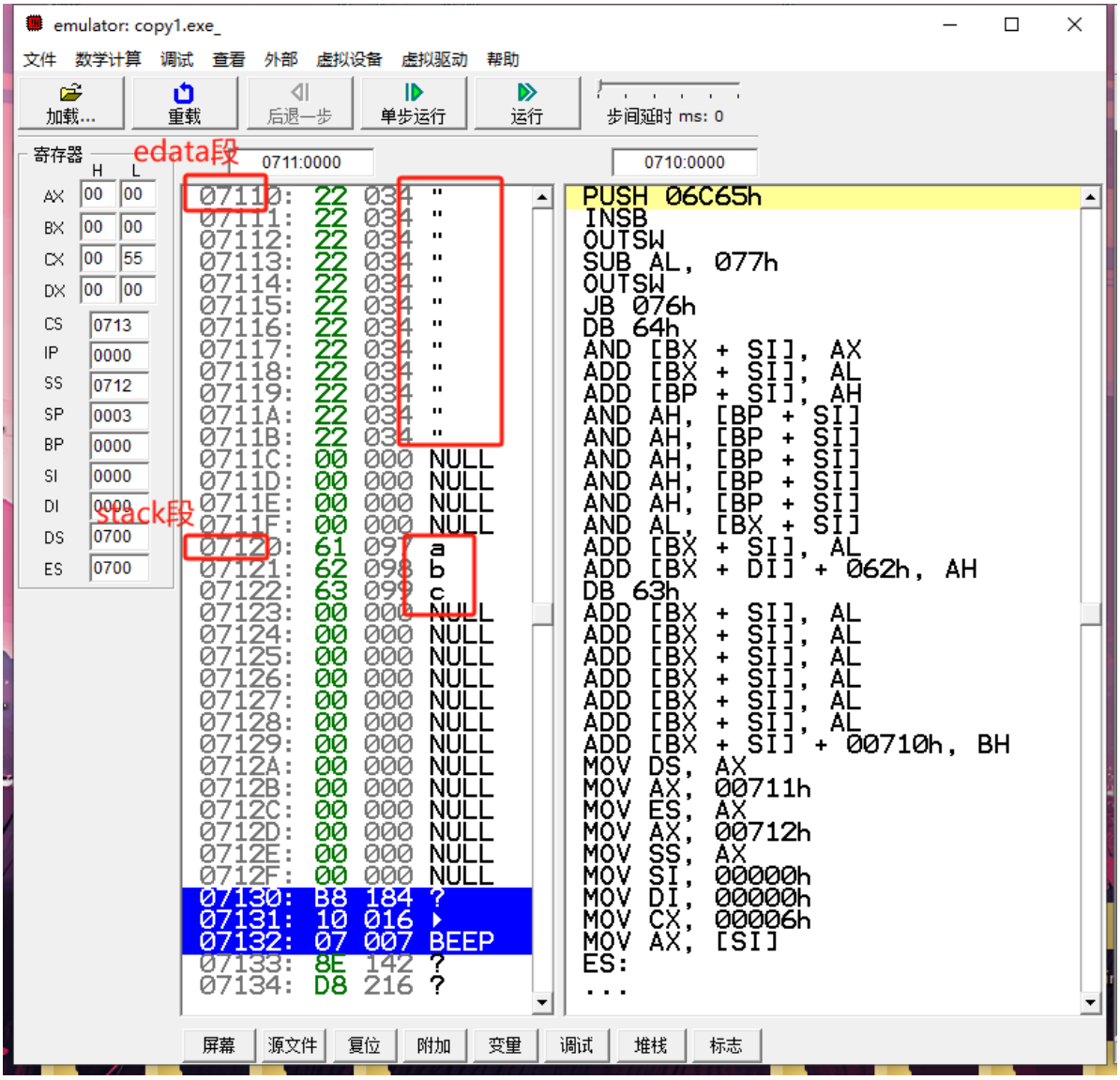
data段（0710h）的内容如下：



注意，左边的是原始的十六进制内容，也就是68，65，...，对应字符串就是h，e，....

右边是强行翻译为指令，没有意义，不用看

edata段（0711h）和stack段（0712h）的内容如下：



分别是12个空格和abc

一些补充说明：

> 定义的段名，如code，data，stack，edata，编译器会自动分配相应的地址，在程序中可以随便使用段名，编译器编译后
> 会自动将段名翻译为相应的地址

> 在每段中无论是定义变量使用的标号（比如数据段写入str1这个字符串），还是定义函数使用的标号（如代码段写入main函数），str1和main这些标号在编写程序时都可以直接当成地址使用，因为编译器会自动将他们翻译为相应的地址

3.汇编程序寄存器使用指南

主要就是编写程序时，对一些寄存器的固定用法，

1. 所有常量可以直接MOV到通用寄存器和内存单元，但是不能直接MOV到段寄存器
2. 常量MOV到内存单元时，需要指定大小，但是从寄存器MOV到内存单元中，不需要指定大小
3. ax寄存器通常用于计算，以及和内存交换数据
4. bx寄存器默认用于data段，作为基址，si，di寄存器作为变址
5. cx寄存器和dx寄存器主要用于loop循环次数计数和变量累加
6. sp寄存器和bp寄存器主要用于管理函数栈，默认用于栈段，也使用si，di寄存器作为变址

关于进一步如何编写循环程序，条件逻辑程序，函数，中断程序，见后面的小节