

****

《汇编语言》实验报告

（四）

**姓　　名 邹济帆**

**学　　号 37220222203904**

**学 院 信息学院**

**专 业 数字媒体技术**

**2023年 10 月**

# 实验目的

#### 1）基于已学习的第三章以及第四章内容，通过分支与循环程序设计，学习如何运用标号、变量伪指令以及程序设计的技巧完成具体的任务；

#### 2）利用debug调试程序，进一步理解分支与循环程序设计的特点；

# 实验内容

# (2) 根据下述情况，分别编写程序，记录 BX 中 1 的个数（需要考虑 BX 中二进制 串的特殊情况），要求如下：

# ⚫ 循环次数已知

# ⚫ 循环次数未知

# (3) 按照下列要求，编写相应程序段。

# 1) 起始地址为 string 的主存单元中存放一个字符串（长度大于 6），把该字符串中的第 1 个和第 6 个字符（字节量）传送给 DX 寄存器；

# 2) 从主存 buffer 开始的 4 字节中保存了 4 个非压缩 BCD 码，现按低（高） 地址对低（高）位的原则，将他们合并到 DX 中。

# 3) 假设从 B800H:0 开始存放有 100 个 16 位无符号数，编程求它们的和，并 存在 DX.AX 中

# 4) 一个 100 字节元素的数组首地址为 array，将每个元素减 1（不考虑溢出）。

# (4) 把内存中从 PACKED 开始的 10 个字节单元中的 16 位压缩 BCD 数转换成非压 缩 BCD 数，并把结果存放在 UNPACKED 开始的 20 个字节单元中；将下列代 码补充完整，并且自己定义 PACKED 中的数据，将 UNPACKED 中的结果展示 出来。 MOV DX, \_\_\_\_\_ MOV CL, \_\_\_\_\_\_ MOV SI, 0 MOV DI, \_\_\_ CONVERT: MOV AL, [SI+PACKED] MOV AH, AL AND AL, 0FH \_\_\_\_\_\_\_\_\_\_ MOV [DI+UNPACKED], \_\_\_\_ ADD DI, \_\_\_\_ \_\_\_\_\_\_ DEC DX JNZ CONVERT

# (5) 给定一个有序数组（均小于 FFH 例如 02H, 07H, 0BH, 0FH, 13H, 1CH, 24H, 39H, 40H, 57H, 68H）和一个目标值（例如 79H），请判断数组中是否含有两个数的 和为目标值，请设计一个算法，将时间复杂度控制在 O(n)，编程实现并验证 你的算法。

# 实验结果

（2）循环次数已知时，设置bx初始值为4637H，将BX向右移位16次。

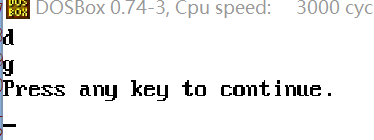
循环次数未知时，设置bx的初值为1145H，将BX向右移位直到BX等于0.



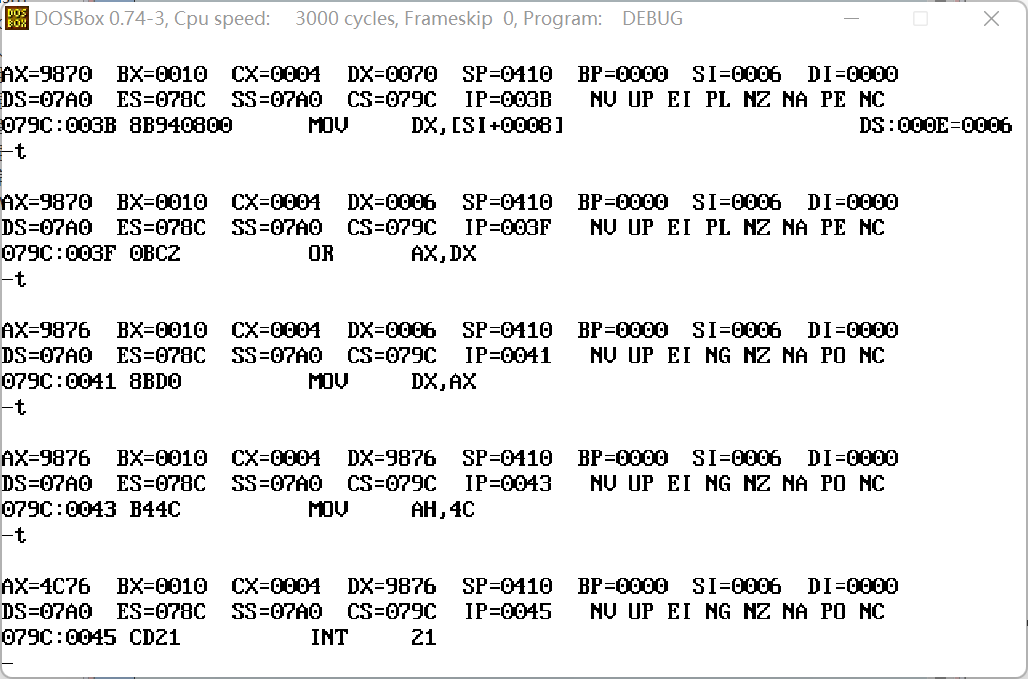
根据结果显示一共移动了13位。

（3）

1）初始化字符串为“debugging”，输出结果如下：



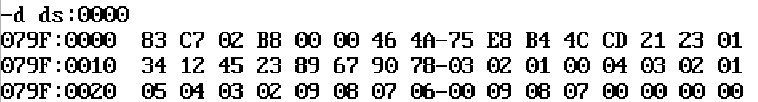
2）保存的四个压缩BCD码为0009，0008，0007，0006，debug结果如下：



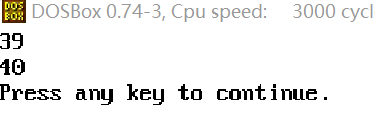
3）由于未初始化数据，不做展示，可以查看代码注释。

4）由于未初始化数据，不做展示，可以查看代码注释。

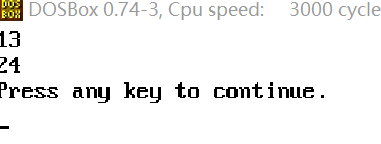
（4）初始化的非压缩bcd码为0123h,1234h,2345h,6789h和7890h,debug结果如下：



（从第2行横杠后方开始为转换后的，第一行倒数第2位到第二行横杠前为转换前的）

（5）该算法返回组成目标数值的两个数(数组为题干例子)。当目标数值为79时：

当目标数为37时：



# 实验分析与总结

虽然可以，但是本次实验没有使用子程序的必要。所有的程序都可以通过简单的分支和循环来完成。其中，有次数循环可以使用loop指令，并且通过CX记录循环次数；无固定次数循环可以使用无条件转移指令JMP，并且在循环体中定义跳出语句。分支结构则更加简单，可以通过多个有条件转移指令JCC完成。

值得注意的是，在本次实验中设置标志来判断是否转移的CMP指令非常常用，说明了该指令的重要性。