

**课 程 实 验 报 告**

**课程名称： 汇编语言程序设计实验**

**实验名称： 实验二 程序优化**

**实验时间： 2019-4-1，18:30-21:50 实验地点： 南一楼803室95号实验台**

**指导教师： 许向阳**

**专业班级：计算机科学与技术201703班**

**学 号： U201714609 姓 名： 李慧骏**

**同组学生： 无 报告日期： 2019年 4 月 2日**

**原创性声明**

  本人郑重声明：本报告的内容由本人独立完成，有关观点、方法、数据和文献等的引用已经在文中指出。除文中已经注明引用的内容外，本报告不包含任何其他个人或集体已经公开发表的作品或成果，不存在剽窃、抄袭行为。

特此声明！

学生签名：

日期：2019.4.2

成绩评定

|  |  |  |
| --- | --- | --- |
| 实验完成质量得分（70分）（实验步骤清晰详细深入，实验记录真实完整等） | 报告撰写质量得分（30分）（报告规范、完整、通顺、详实等） | 总成绩（100分） |
|  |  |  |

指导教师签字：

                    日期：

**目录**

[**1 实验目的与要求 3**](#_Toc5137475)

[**2 实验内容 3**](#_Toc5137476)

[**3 实验过程 4**](#_Toc5137477)

[**任务1 4**](#_Toc5137478)

[**3.1.1 设计思想 4**](#_Toc5137479)

[**3.1.2 流程图 5**](#_Toc5137480)

[**3.1.3 程序代码 5**](#_Toc5137481)

[**3.1.4 实验步骤 9**](#_Toc5137482)

[**3.1.5 程序运行与测试 10**](#_Toc5137483)

[**任务2 11**](#_Toc5137484)

[**3.2.1乘法指令进行优化 11**](#_Toc5137485)

[**3.2.2 寻址方式优化 12**](#_Toc5137486)

[**3.2.3 子程序调用优化 12**](#_Toc5137487)

[**3.2.4 优化总结 12**](#_Toc5137488)

[**4 实验小结 13**](#_Toc5137489)

# 实验目的与要求

1. 了解程序计时的方法以及运行环境对程序执行情况的影响。
2. 熟悉汇编语言指令的特点，掌握代码优化的基本方法

# 实验内容

**任务1. 观察多重循环对CPU计算能力消耗的影响**

应用场景介绍：以实验一任务4的背景为基础，只要有一个顾客访问网店中的商品，系统就需要计算一遍所有商品的推荐度（**本次实验都要按照此需求计算推荐度**），然后再处理顾客实际购买的商品的信息。现假设在双十一零点时，SHOP网店中的“Bag”商品共有m件，有m个顾客几乎同时下单购买了该商品。请模拟后台处理上述信息的过程并观察执行的时间。

上述场景的后台处理过程，可以理解为在同一台电脑上有m个请求一起排队使用实验一任务4的程序。为了观察从第1个顾客开始进入购买至第m个顾客购买完毕之间到底花费了多少时间，我们让实验一任务4的功能三调整后的代码重复执行m次，通过计算这m次循环执行前和执行后的时间差，来感受其影响。功能三之外的其他功能不纳入到这m次循环体内（但可以保留不变）。

**调整后的功能三的描述：**

（1）提示用户输入要购买的商品名称（比如“Bag”）。

（2）计算SHOP中所有商品的推荐度。

（3）在SHOP中找到顾客购买的商品（比如“Bag”，若未能找到该商品，回到（1）重新输入。若只输入回车，则回到功能一（1））。

（4）判断该商品已售数量是否大于等于进货总数，若是，则回到功能一（1），否则将已售数量加1。

（5）回到功能三（1）。

请按照上述设想修改实验一任务4的程序，并将m和n值尽量取大（比如大于1000，具体数值依据实验效果来改变，逐步增加到比较明显的程度，比如秒级的时间间隔。另外，也可以把定义“Bag”的位置放在所有商品的最后，使得搜索它的时间变长），以得到较明显的效果。

**提示:** 学校汇编教学网站的软件下载中提供了显示当前时间“秒和百分秒”的子程序。若在m次循环前调用一下该子程序，m次循环执行完之后再调用一下该子程序，就能在屏幕上观察并感受到执行循环前后的时间差（时间差值需要自行手工计算，当然，你也可以选用网站上另一个计时程序，它是可以帮你计算好差值的）。注意，由于虚拟机环境下CPU会被分时调度，故该时间差值会因计算机运行环境、状态以及虚拟机的设置参数的不同而不同。

**任务2.** 对任务1中的汇编源程序进行优化

优化工作包括代码长度的优化和执行效率的优化，本次优化的重点是执行效率的优化。请通过优化m次循环体内的程序，使程序的执行时间尽可能减少10%以上（注意，在编写任务1的程序时，尽量不要考虑代码优化的问题）。

**优化方法提示：**首先是通过选择执行速度较快的指令来提高性能，比如，把乘除指令转换成移位指令、加法指令等；其次，内循环体中每减少一条指令，就相当于减少了m\*n条指令的执行时间，需要仔细斟酌；第三，在寻址方式中尽量把16位寄存器换成32位寄存器，能有**更多的机会和技巧**提高指令执行效率。

# 实验过程

## 任务1

### 设计思想

在实验一任务五的基础上进行代码修改，利用循环语句在用户输入了相应商品后进行m次循环，每次循环都将n种商品的推荐度完成计算。通过修改n与m的值检测改变循环次数对CPU的资源差异影响。

### 流程图

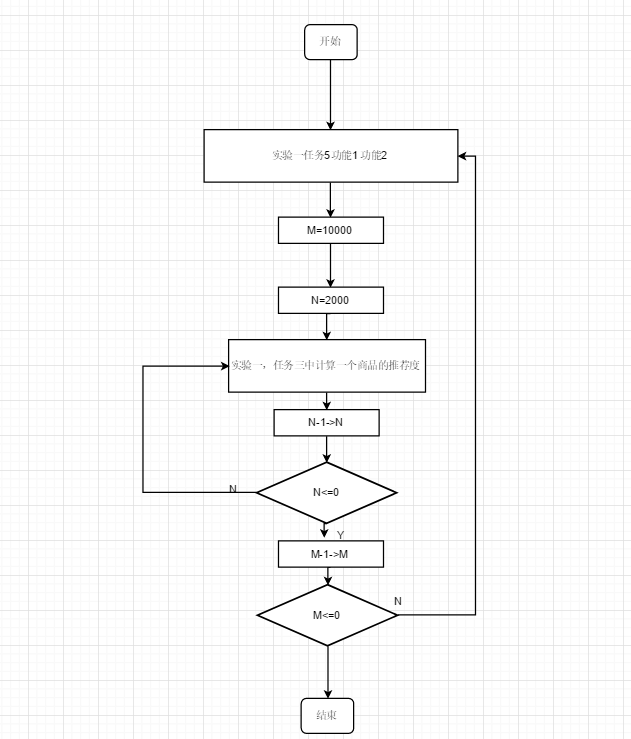


图2.1 流程图

### 3.1.3 程序代码

.386

DATA SEGMENT USE16

BUF DB 50

DB 0

DB 50 DUP(0)

BUF1 DB 50

DB 0

DB 50 DUP(0)

CRLF1 DB 0DH,0AH,"WELCOME TO HUTS SHOP!$"

CRLF2 DB 0DH,0AH,"INPUT USE'S NAME!$"

CRLF3 DB 0DH,0AH,"INPUT PASSWORD!$"

CRLF4 DB 0DH,0AH,"INPUT THE GOODS!$"

CRLF5 DB 0DH,0AH,"HELLO BOSS!$"

CRLF6 DB 0DH,0AH,"HELLO CUSTOMER!$"

CRLF7 DB 0DH,0AH,"HELLO !$"

CRLF8 DB 0DH,0AH,"FIND GOODS !$"

CRLF9 DB 0DH,0AH,"THIS GOODS IS A !$"

CRLF10 DB 0DH,0AH,"THIS GOODS IS B !$"

CRLF11 DB 0DH,0AH,"THIS GOODS IS C !$"

CRLF12 DB 0DH,0AH,"THIS GOODS IS F !$"

BNAME DB "LI HUIJUN",0

BPASS DB "TEST",0,0

N EQU 1000

M EQU 1000

SNAME DB "SHOP",0,0,0,0,0,0

GA1 DB "PEN" ,7 DUP(0),10

DW 35,56,2000,25,?

GA2 DB "BOOK",6 DUP(0),9

DW 12,30,2000,5,?

GAN DB N-2 DUP("TEMP-VALUE",8,15,0,20,0,30,0,2,0,?,?)

COUNT DW 5000

DATA ENDS

STACK SEGMENT USE16 STACK

DB 500 DUP(0)

STACK ENDS

CODE SEGMENT USE16

ASSUME DS:DATA,SS:STACK,CS:CODE

；增加的显示时间子程序

DISPTIME PROC ;显示秒和百分秒，精度为55MS。(未保护AX寄存器)

LOCAL TIMESTR[8]:BYTE ;0,0,'"',0,0,0DH,0AH,'$'

PUSH CX

PUSH DX

PUSH DS

PUSH SS

POP DS

MOV AH,2CH

INT 21H

XOR AX,AX

MOV AL,DH

MOV CL,10

DIV CL

ADD AX,3030H

MOV WORD PTR TIMESTR,AX

MOV TIMESTR+2,'"'

XOR AX,AX

MOV AL,DL

DIV CL

ADD AX,3030H

MOV WORD PTR TIMESTR+3,AX

MOV WORD PTR TIMESTR+5,0A0DH

MOV TIMESTR+7,'$'

LEA DX,TIMESTR

MOV AH,9

INT 21H

POP DS

POP DX

POP CX

RET

DISPTIME ENDP

START: MOV AX,DATA

MOV DS,AX

;网店名称

LEA DX,CRLF1

MOV AH,9

INT 21H

.

. 输入用户名、密码、商品名同实验一任务五

.

.

;计算推荐度

GO2 : CALL DISPTIME

GO1:

MOV CX,1000

GO: LEA BX ,GA1

MOV DI,21

MOV AL ,[BX+DI-10];折扣

CBW

MOV SI,AX

MOV AX,[BX+DI-7]

MUL SI

MOV SI,10

MOV DX,0

DIV SI

MOV SI,AX;实际销售价格

MOV AX ,[BX+DI-9];进货价

MOV DX,0

DIV SI;进货价/实际销售价

MOV SI,AX

MOV AX ,[BX+DI-3];已售数量

MOV BP,[BX+DI-5];进货总量

MOV DX,0

DIV BP;已售数量/进货总量

ADD AX,SI

MOV SI,64

MUI SI

MOV SI,2

MUI SI

MOV [BX+DI-1],AX

ADD DI,21

MOV AX ,[BX+DI-3];已售数量

DEC AX

MOV [BX+DI-3],AX

CMP [BX+DI-5],AX

JNZ NO

LOOP GO

DEC COUNT

JNZ GO1

CALL DISPTIME

MOV AH,4CH

INT 21H

CODE ENDS

END START

### 实验步骤

1.在notepad上完成代码编译，利用masm生成obj文件、利用link生成exe文件，利用td进行汇编。

2.程序启动正常，如图2.2所示。

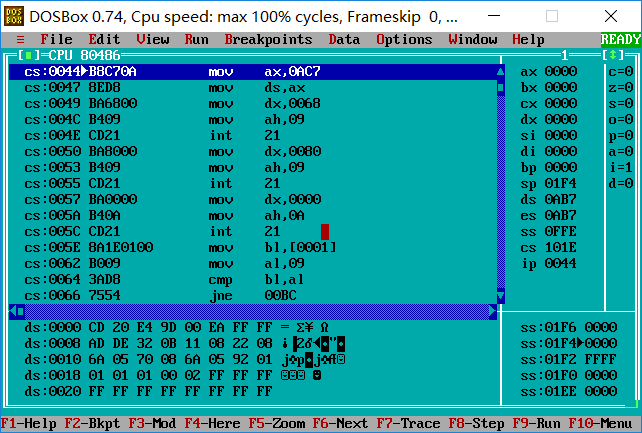


图2.2 TD运行程序正常

### 3.1.5 程序运行与测试

1. 尝试改变循环次数观察运行时间

先将访问顾客数m设置为5000，商品数n设置为1000，运行程序所花费时间如图2.3所示。由进入循环前与退出循环后的两个时间段可知，m=5000,n=1000时，花费50毫秒。

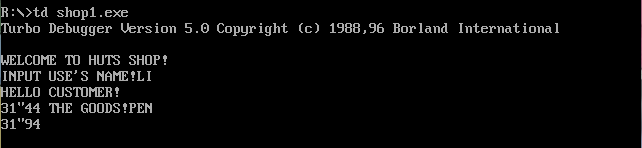


图2.3 未扩大前所需时间

尝试将m扩大为10000，n扩大为2000，运行程序所花费时间如图1.4所示。由进入循环前与退出循环后的两个时间段可知，m=10000,n=2000时，花费186毫秒。

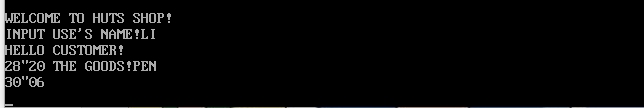


图2.4 扩大用户与商品数后所需时间

2. 尝试在循环中加入9号调用观察运行时间

在m=10000,n=2000的情况下，在循环中利用9号调用输出“WELCOME TO HUTS SHOP”语句，结果如图2.5所示。在程序运行了一分钟以后任然没有完成m\*n次“WELCOME TO HUTS SHOP”的输出，可见在循环中增加一句9号调用所花费的时间远大于增加循环次数的影响。

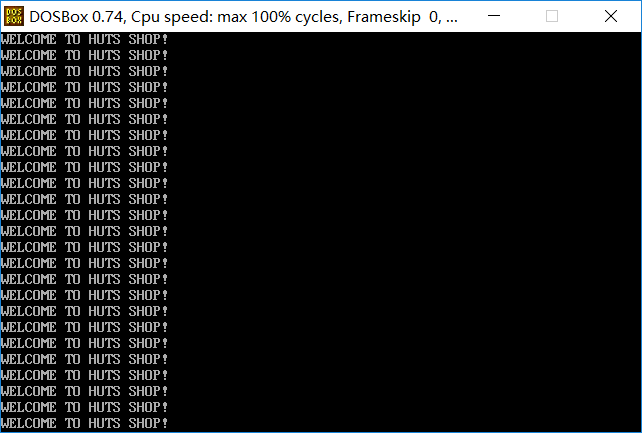


图2.5 尝试在循环中加入9号调用观察运行时间

## 任务2

### 3.2.1乘法指令进行优化

之前考虑到128的溢出问题，将乘以128替换为分别乘以64与2，现为提升运算效率，将其更换为位移运算，如图2.6所示。在m=10000,n=2000时运行程序，所花费时间如图2.7所示。利用位移代替乘法后花费时间为181毫秒，与图2.4所测的乘法运算186毫秒相比并无明显提速。

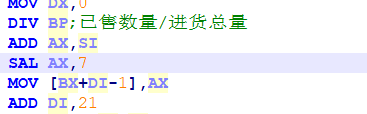


图2.6 用位移运算代替乘法运算

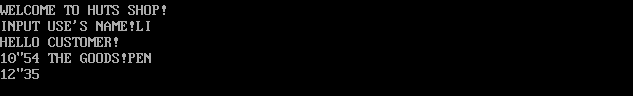


图2.7 位移运算替代后的时间

### 3.2.2 寻址方式优化

尝试优化寻址方式的书写，将“MOV AX ,[BX+DI-3];已售数量 INC AX MOV [BX+DI-3],AX”三句修改为一句指令，在m=10000,n=2000情况下运行，花费时间如图2.8所示。计算可得花费150毫秒，有着明显优化。



图2.8 优化传址方式后时间

### 3.2.3 子程序调用优化

由于程序在循环中并没有调用子程序，固将显示时间的DISPTIME子程序删除输出时间功能后在循环中调用，运行时间如图2.9所示。对比3.2.2优化后的150毫秒运行时间，添加了没有9号调用的子程序后时间增加到了170毫秒，可见子程序调用对于运行效率有着明显影响。

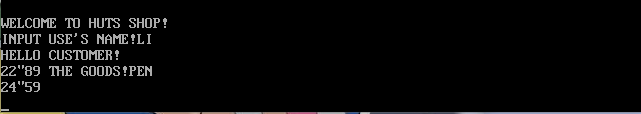


图2.9 增加子程序调用后时间

### 3.2.4 优化总结

通过3.2.1与3.2.2的优化后，程序运行时间由186毫秒减少到了150毫秒，优化率达到了24%。同时发现由位操作指令代替乘法指令对于程序的优化并不明显。而在循环中调用子程序与简化指令等操作对于程序的优化效果更佳明显。

# 实验小结

任务2的优化总结已经在3.2.4中表述，再此 赘述。

对于任务1的体会主要有一下几点。

一、溢出问题，最开始并没有考虑到在计算时产生溢出的情况而导致在TD中出现FE而使程序进入了死循环。解决操作是并没有更换寄存器而是分别乘以了64与2来代替直接乘以128，解决溢出问题。

二、跳转问题，忽视了LOOP指令跳转收到了距离的限制而报错，后选用JNZ等语句进行代替LOOP，解决的跳转的距离问题。

三、寻址方式问题，由于对于之前寻址方式掌握的不够理想，加上遗忘了很多，导致在编写寻址方式时频频出错，通过本次实验也是再一次加强了我对于寻址方式的训练。