

**课 程 实 验 报 告**

**课程名称： 汇编语言程序设计实验**

**实验名称： 实验五 汇编语言与C语言混合编程**

**实验时间： 2016-4-19，14：00-17：30 实验地点： 南一楼804室74号实验台**

**指导教师： 张勇 专业班级：计算机科学与技术201409班**

**学 号： U201414813 姓 名： 唐礼威**

**同组学生： 无 报告日期： 2016年 4月 23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：

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 1](#_Toc451727849)

[2 实验内容 1](#_Toc451727850)

[3 实验过程 2](#_Toc451727851)

[3.1 任务1 2](#_Toc451727852)

[3.1.1 设计思想及存储单元分配 2](#_Toc451727853)

[3.1.2 流程图 3](#_Toc451727854)

[3.1.3 源程序 3](#_Toc451727855)

[3.1.4 实验步骤 6](#_Toc451727856)

[3.1.5 实验记录 6](#_Toc451727857)

[3.2 任务2 7](#_Toc451727858)

[3.2.1 设计思想及存储单元分配 7](#_Toc451727859)

[3.2.2 流程图 8](#_Toc451727860)

[3.2.3 源程序 8](#_Toc451727861)

[3.2.4 实验步骤 11](#_Toc451727862)

[3.2.5 实验记录 11](#_Toc451727863)

[4 体会 11](#_Toc451727864)

[参考文献 12](#_Toc451727865)

# 实验目的与要求

(1) 掌握汇编语言程序与C语言程序混合编程的方法；

(2) 熟悉C编译器的基本优化方法;

(3) 了解C语言编译器的命名方法，主、子程序之间参数传递的机制。

# 实验内容

任务1：在C 语言程序中嵌入汇编语言指令语句序列

对于实验二的程序进行改造，主控程序、以及输入输出等功能用C语言实现，学生姓名搜索和成绩计算用C程序中嵌入汇编指令语句序列的方式实现。

任务2：在C语言程序中调用 汇编语言实现的函数

对于实验二的程序进行改造，主控程序、以及输入输出等功能用C语言实现，学生姓名搜索和成绩计算用独立的汇编语言子程序的方式实现； 在C语言程序中调用汇编语言子程序。

要求：

（1）在不同的C语言开发环境中实现与汇编语言程序的混合编程，其操作方法有可能是不同的。请大家选择自己熟悉的C语言开发环境并查找相关的资料完成本实验。

（2）在实验报告中，比较详细的给出你的开发环境及其实现方法。

（3）观察C语言编译器中对各种符号的命名规则（指编译器内部可以识别的命名规则，比如，符号名前面是否加下划线“\_”，等），主、子程序之间参数传递的机制，通过堆栈传递参数后堆栈空间回收的方法。

（4） 对混合编程形成的执行程序，用调试工具观察由C语言形成的程序代码与由汇编语言形成的程序代码之间的相互关系，包括段、偏移的值，汇编指令访问C的变量时是如何翻译的，等。

（5） 请尝试在C语言源程序中不合理地嵌入汇编语言的指令语句，达到破坏C语言程序的正确性的目的。比如，在连续的几条C语言语句中间加入一条修改AX寄存器的汇编指令语句，而AX的内容在此处本不该被修改，这样就可观察到破坏C语言程序正确性的效果（该项实验表明：在C语言程序中，若不考虑上下语句翻译成怎样的机器码而随意嵌入汇编指令语句时，有可能存在出错的风险）。

（6） 观察C编译器的优化策略对代码的影响。

（7） 通过调试混合编程的程序，体会与纯粹汇编语言编写的程序的调试过程的差异。

（8）通过本次实验，希望大家明白：不同的编程语言是可以协同解决一个问题的，而且可以利用不同语言的特点来更好地解决问题；利用汇编语言的知识，能够更好地理解高级语言的内部处理原理与策略，为编写更好的C语言程序、用好C编译器提供支持。

# 实验过程

## 任务1

### 设计思想及存储单元分配

任务：在C 语言程序中嵌入汇编语言指令语句序列

对于实验二的程序进行改造，主控程序、以及输入输出等功能用C语言实现，学生姓名搜索和成绩计算用C程序中嵌入汇编指令语句序列的方式实现。

整体结构功能通过一个do-while循环来实现，而循环的条件根据使用者输入“Y”或“N”来判断。在整个循环的过程中，学生姓名搜索和成绩计算用C语言内嵌的汇编指令来实现。

1.结构设计：

struct Stu{

char name[12]; //储存姓名

short score[4]; //储存成绩

};

2.存储单元分配

Name:分配12个字节来存储名字

Score:分配1个短整型数组，数组包含4个元素，分别为语文、数学、英语和平均成绩。

### 流程图



### 源程序

#include<stdlib.h>

#include<stdio.h>

#include<string.h>

struct Stu{

char name[12]; //储存姓名

short score[4]; //储存成绩

};

int main() {

struct Stu info[3];

char name[12]; //输入的姓名

int i, j; //计数器

char op;

//初始化三个学生信息用于测试

strcpy(info[0].name,"zhangsan\_a");

info[0].score[0] = 86;

info[0].score[1] = 82;

info[0].score[2] = 88;

info[0].score[3] = 0;

strcpy(info[1].name, "zhangsan\_b");

info[1].score[0] = 48;

info[1].score[1] = 79;

info[1].score[2] = 46;

info[1].score[3] = 0;

strcpy(info[2].name, "zhangsan\_c");

info[2].score[0] = 96;

info[2].score[1] = 92;

info[2].score[2] = 98;

info[2].score[3] = 0;

do{

printf("Inlut the name of the student:");

gets(name);

\_asm {

mov ecx, 3

lea edi, info[0];

LOAPI:

lea esi, name

mov edx, 12

LOAPJ :

mov al, byte ptr[esi]

mov bl, byte ptr[edi]

cmp al, bl

jnz NEXT

cmp al, '\0'

jz FIGURE

dec edx

inc edi

inc esi

jmp LOAPJ

NEXT :

dec ecx

jz REINPUT

add edx, 8

add edi, edx

jmp LOAPI

FIGURE :

add edi, edx

mov al, byte ptr[edi]

mov ah, 0

add ax, ax

add ax, ax

mov dl, byte ptr[edi + 2]

mov dh, 0

add ax, dx

add ax, dx

mov dl, byte ptr[edi + 4]

mov dh, 0

add ax, dx

mov bl, 7

div bl

mov ah, 0

mov word ptr[edi + 6], ax

}

for (i = 0; i < 3; i++)

{

if (!strcmp(info[i].name, name))

{

printf("%s\t", info[i].name);

for (j = 0; j < 4; j++)

{

printf("%d\t", info[i].score[j]);

}

printf("\n");

}

}

\_asm{

REINPUT:

}

printf("Input Y to exit,N continue...");

op = getchar();

getchar();

} while (op == 'Y' || op == 'y');

for (i = 0; i < 3; i++) {

printf("%s\t", info[i].name);

for (j = 0; j < 4; j++){

printf("%d\t", info[i].score[j]);

}

printf("\n");

}

getchar();

return 0;

}

### 实验步骤

1. 使用Visual C++录入源程序，编译，链接，运行。

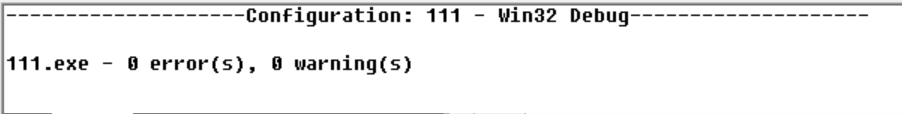
2. 观察提示信息，若出错，则用编辑程序修改错误，修改后再重新编译

3. 执行该程序。观察程序运行结果。

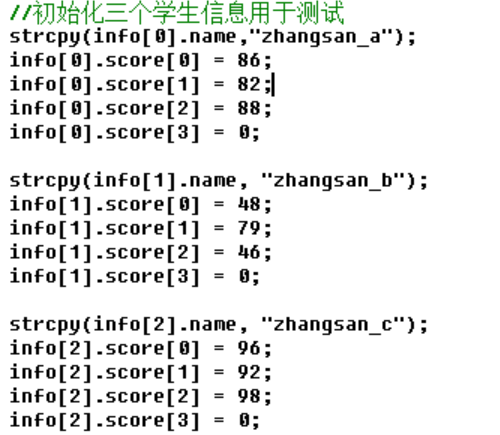
### 实验记录

（包括实验条件、输入/输出、错误和修改等信息的记录）

1. 实验环境条件：Visual C++；Visual C++窗口；
2. 编译源程序时没有发生异常。
3. 连接过程没有发生异常。



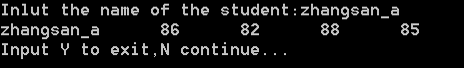
1. 输入提前录入的三个学生姓名。



1. 运行目标程序



1. 根据程序提示输入查找的学生姓名



输入提前录入的学生姓名，输出该学生的各项成绩，并提示是否继续。

## 任务2

### 设计思想及存储单元分配

任务：在C语言程序中调用 汇编语言实现的函数

对于实验二的程序进行改造，主控程序、以及输入输出等功能用C语言实现，学生姓名搜索和成绩计算用独立的汇编语言子程序的方式实现； 在C语言程序中调用汇编语言子程序。

整体结构功能通过一个do-while循环来实现，而循环的条件根据使用者输入“Y”或“N”来判断。在整个循环的过程中，C语言通过调用汇编语言子程序，来达到学生姓名搜索和成绩计算的功能。而这两个功能则独立利用汇编语言子程序的方式来实现。

1.结构设计：

struct Stu{

char name[12]; //储存姓名

short score[4]; //储存成绩

};

2.存储单元分配

Name:分配12个字节来存储名字

Score:分配1个短整型数组，数组包含4个元素，分别为语文、数学、英语和平均成绩。

### 流程图

无

### 源程序

汇编语言子程序

.386

.model flat, c

.code

public dif

dif proc par1:dword,par2:dword

mov ecx, 1000

mov edi, par1;

LOAPI:

mov esi, par2

mov edx, 12

LOAPJ:

mov al, byte ptr[esi]

mov bl, byte ptr[edi]

cmp al, bl

jnz NEXT

cmp al, 0

jz FIGURE

dec edx

inc edi

inc esi

jmp LOAPJ

NEXT:

dec ecx

jz REINPUT

add edx, 8

add edi, edx

jmp LOAPI

FIGURE:

add edi, edx

mov al, byte ptr[edi]

mov ah, 0

add ax, ax

add ax, ax

mov dl, byte ptr[edi + 2]

mov dh, 0

add ax, dx

add ax, dx

mov dl, byte ptr[edi + 4]

mov dh, 0

add ax, dx

mov bl, 7

div bl

mov ah, 0

mov word ptr[edi + 6], ax

REINPUT:

ret

dif endp

end

C语言主程序

#include<stdlib.h>

#include<stdio.h>

#include<string.h>

struct Stu{

char name[12]; //储存姓名

short score[4]; //储存成绩

};

extern void dif(struct stuinfo \*, char \*);

int main() {

struct Stu info[3];

char name[12]; //输入的姓名

int i, j; //计数器

char op;

//初始化三个学生信息用于测试

strcpy(info[0].name,"zhangsan\_a");

info[0].score[0] = 86;

info[0].score[1] = 82;

info[0].score[2] = 88;

info[0].score[3] = 0;

strcpy(info[1].name, "zhangsan\_b");

info[1].score[0] = 48;

info[1].score[1] = 79;

info[1].score[2] = 46;

info[1].score[3] = 0;

strcpy(info[2].name, "zhangsan\_c");

info[2].score[0] = 96;

info[2].score[1] = 92;

info[2].score[2] = 98;

info[2].score[3] = 0;

do{

printf("Inlut the name of the student:");

gets\_s(name, 12);

dif(info, name);

for (i = 0; i < 3; i++){

if (!strcmp(info[i].name, name)){

printf("%s\t", info[i].name);

for (j = 0; j < 4; j++){

printf("%d\t", info[i].score[j]);

}

printf("\n");

}

}

printf("Input Y to exit,N continue...");

op = getchar(); getchar();

} while (op == 'Y' || op == 'y');

for (i = 0; i < 3; i++) {

printf("%s\t", info[i].name);

for (j = 0; j < 4; j++){

printf("%d\t", info[i].score[j]);

}

printf("\n");

}

getchar();

return 0;

}

### 实验步骤

略

### 实验记录

见任务一

# 体会

通过这次实验，更加深刻地认识到了汇编语言与高级语言之间的关系，懂得如何将汇编语言与C语言程序混合编程的方法，更加清楚了主、子程序之间参数传递的机制。为今后混合编程打下了基础，相信在今后的编程实践中会变得熟练很多。

# 参考文献

[1] 王元珍等.80x86汇编语言程序设计.版本(第1版)

[2] 王晓虹等.汇编语言程序设计教程.版本(第1版)