

实验报告

实验名称	实验二 循环展开与指令流水		
实验教室	丹青 911	实验日期	2022 年 11 月 10 日
学 号	2019210173	姓 名	刘思远
专业班级	奥林学院计算机科学与技术 04 班		
指导教师	卢洋		

东北林业大学
信息与计算机科学技术实验中心

一、 实验目的

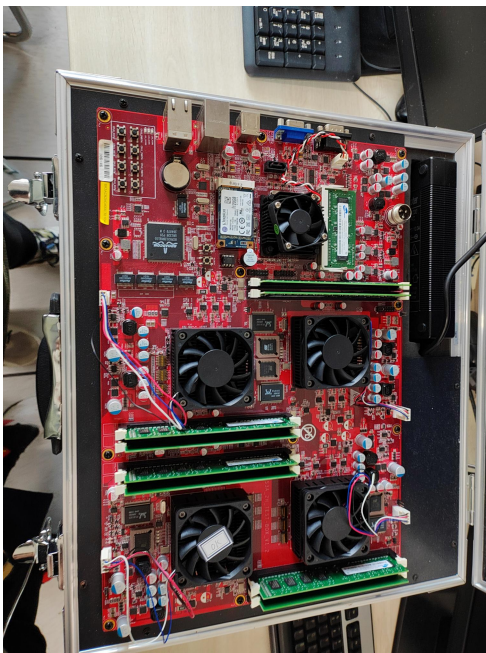
1. 加深对指令流水理解
2. 掌握利用流水功能的编程技巧

二、 实验环境

多路处理计算器

三、 实验内容及结果

1. 首先，将外接设备接好计算器，并启动计算器



2. 准备好三个程序，分别为

"for.c"-----C 语言：循环数组加法计算

```
/*
 * @Author: NEFU AB-IN
 * @Date: 2022-11-10 10:18:39
 * @FilePath: \undefinedc:\Users\liusy\Desktop\for.c
 * @LastEditTime: 2022-11-10 10:19:16
 */
/* for.c */
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main()
{
    float a[200000];
    int i = 0;
    int duration;
    clock_t start, finish;
    start = clock();
    for (i = 0; i < 200000; i++)
    {
        a[i] = 5.0;
    }
    for (i = 0; i < 200000; i++)
    {
        a[i] = a[i] + 10.0;
    }
    finish = clock();
    duration = (int)(finish - start) * 1000;
    printf("the clock time is: %d\n", duration);
    return 0;
}
```

"for_asm_s.c"----- 嵌入式汇编：无循环展开的数组加法计算

```
/*
```

```

* @Author: NEFU AB-IN
* @Date: 2022-11-10 10:19:35
* @FilePath: \undefinedc:\Users\liusy\Desktop\for_asm_s.c
* @LastEditTime: 2022-11-10 10:20:00
*/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{
    float a[200000];
    int i = 0;
    int x = 4, c, len = 200000;
    int ans;
    int duration;
    clock_t start, finish;

    for (i = 0; i < 200000; i++)
    {
        a[i] = 5;
    }
    start = clock();
    __asm__(
        "li.s $f2, 10.0\n"
        "Loop: \n"
        "lwc1 $f0, 0(%[a])\n"
        "add.s $f0, $f0, $f2\n"
        "swc1 $f0, 0(%[a])\n"
        "addiu %[a], 4 \n"
        "addiu %[ans], 0 \n"
        "addiu %[len], -1\n"
        "bne %[len], $0, Loop\n"
        : [ans] "=r"(ans)
        : [a] "r"(a), [len] "r"(len));
    finish = clock();
    duration = (int)(finish - start) * 1000;
    printf("the clock time is: %d\n", duration);
}

```

“for_asm_open.c”---- 嵌入式汇编：有四重循环展开的数组加法计算

```
/*
 * @Author: NEFU AB-IN
 * @Date: 2022-11-10 10:20:19
 * @FilePath: \undefinedd:\Code\Course\ComputerSystemStructure\实验报告\实验
2\for_asm_open.c
 * @LastEditTime: 2022-11-10 14:11:02
 */
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{
    float a[200000]; // 定义数组变量
    int i = 0;
    int c, len = 200000;
    int ans;
    int duration;
    clock_t start, finish; // 定义时间变量

    for (i = 0; i < 200000; i++)
    {
        a[i] = 5.0;
    }
    start = clock(); // 开始计时
    int iter = 20;
    for (i = 0; i < iter; i++)
    {
        __asm__("li.s $f2, 10.0\n\t" // 采用指令 li.s 对 f2 进行赋值，并赋值为 10.0（随便赋值的），在整数类型里面
                // 是用 li 来赋值，浮点数类型则使用 li.s。
                "Loop: \n\t" // 设置循环
                "lwc1 $f0, 0(%[a])\n\t" // 这一句话是把值赋予给 f0；这里的 a 用于外部参数输入，用%表示: 0(%[a])
                // 指的是对 a 这个外部参数偏移 0 个字节
```

```

        "lwc1 $f6,4(%[a])\n\t" // 这一句话是把值赋予给 f6; 这里的 a 用于外部参数输入, 用%
表示: 4(%[a])

        // 指的是对 a 这个外部参数偏移 4 个字节

        "lwc1 $f10,8(%[a])\n" // 这一句话是把值赋予给 f10; 这里的 a 用于外部参数输入, 用%
表示: 8(%[a])

        // 指的是对 a 这个外部参数偏移 8 个字节

        "lwc1 $f14,12(%[a])\n" // 这一句话是把值赋予给 f14; 这里的 a 用于外部参数输入, 用%
表示: 12(%[a])

        // 指的是对 a 这个外部参数偏移 12 个字节

        "add.s $f4,$f0,$f2\n" // 这一句话是把 f0 和 f2 相加, 并赋值给 f4
        "add.s $f8,$f6,$f2\n" // 这一句话是把 f0 和 f2 相加, 并赋值给 f8
        "add.s $f12,$f10,$f2\n" // 这一句话是把 f0 和 f2 相加, 并赋值给 f12
        "add.s $f16,$f14,$f2\n" // 这一句话是把 f0 和 f2 相加, 并赋值给 f16

        "swc1 $f4,0(%[a])\n" // 保存结果
        "swc1 $f8,4(%[a])\n" // 保存结果
        "swc1 $f12,8(%[a])\n" // 保存结果
        "swc1 $f16,12(%[a])\n" // 保存结果

        "addiu %[len],-4\n" // 循环次数减 4, 因为我们是做了 4 次循环展开

        "addiu %[a],16\n" // 指针偏移 16 位, 因为我们 4 次循环展开, 每个数有 4 个字节,
4*4=16

        "bne %[len],$0,Loop\n" // 判断是否跳出循环

        : [ans] "=r"(ans) // 输出, 这是 asm 编程需要的, 这语句不能删除, 但此程序不需要输
出, 所

        // 以这个 ans 没有实际含义

        : [a] "r"(a), [len] "r"(len) // 外部输入, 把数组和数组长度传入

    );
}

finish = clock(); // 结束计时
duration = (int)(finish - start) / iter;
printf("the clock time is: %d\n", duration);
}

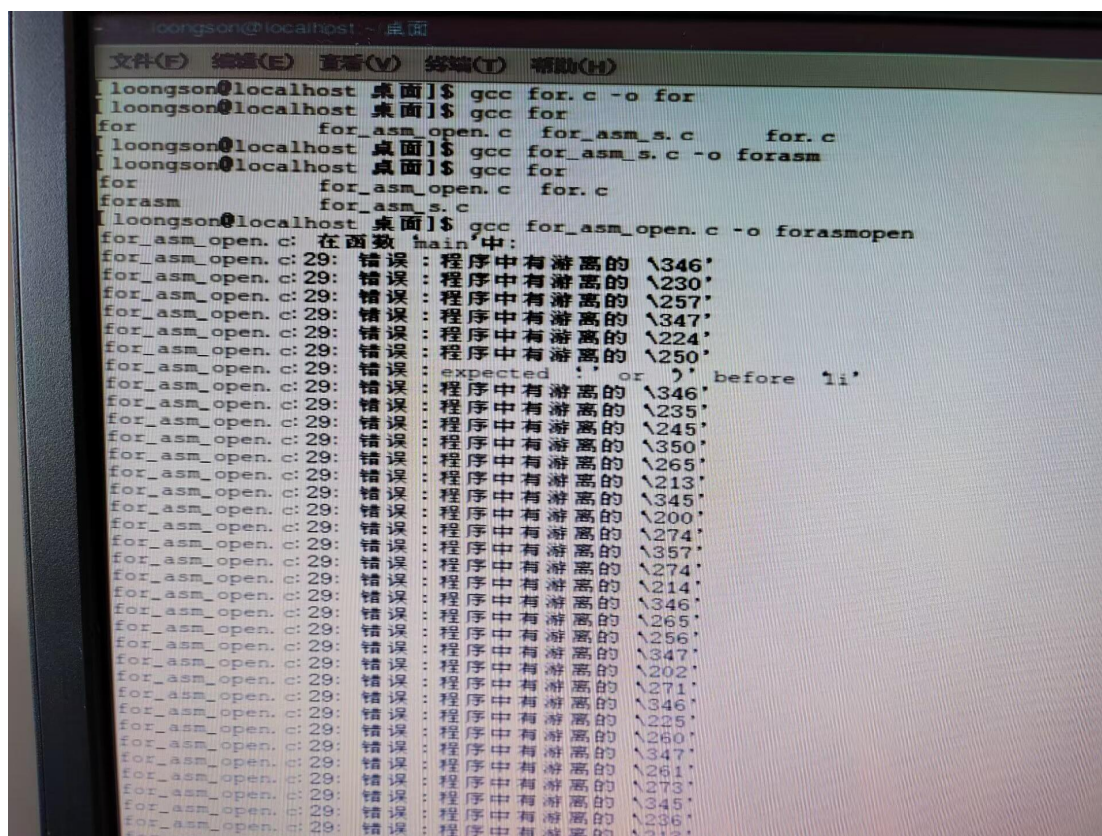
```

开始编译三个源文件, 发现第三个文件有点问题, 进行DEBUG

```
gcc for.c -o for
```

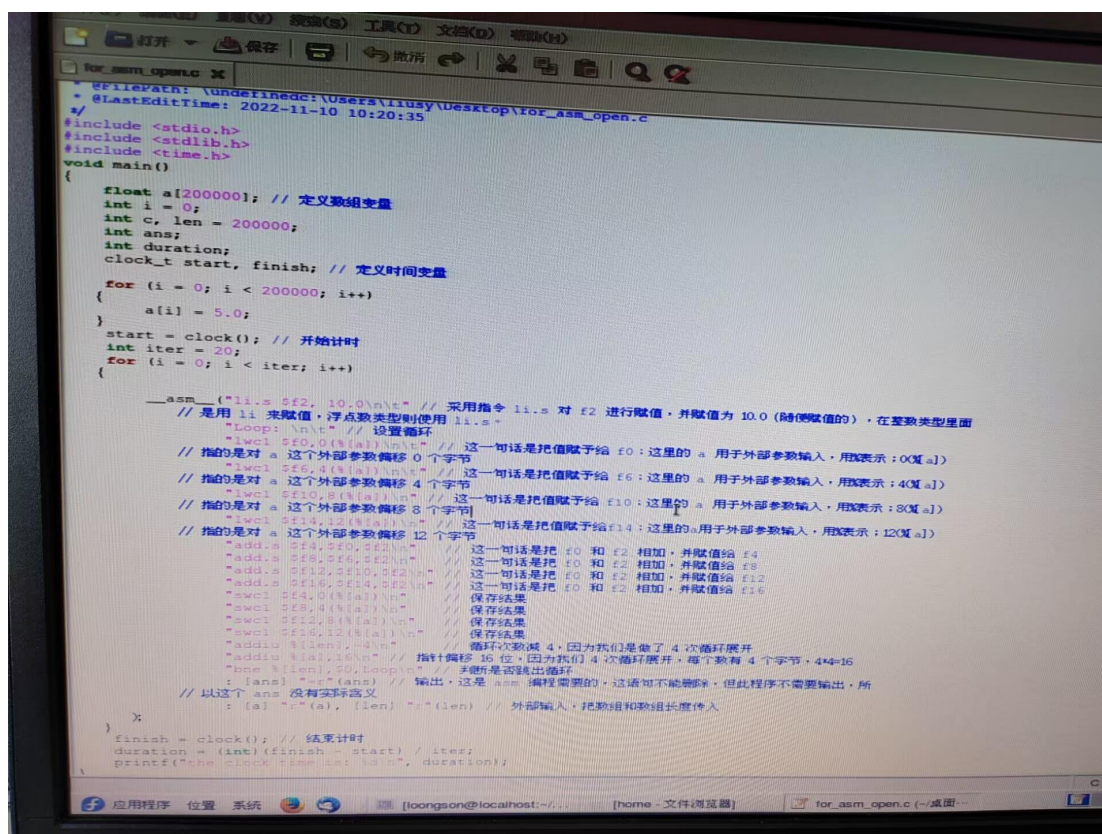
```
gcc for_asm_s.c -o forasm
```


gcc for_asm_open.c -o forasmopen



```
loongson@localhost: ~$ gcc for.c -o for
loongson@localhost: ~$ gcc for
for
for_asm_open.c for_asm_s.c for.c
loongson@localhost: ~$ gcc for_asm_s.c -o forasm
for
for_asm_open.c for.c
forasm
for_asm_s.c
loongson@localhost: ~$ gcc for_asm_open.c -o forasmopen
for_asm_open.c: 在函数 main 中:
for_asm_open.c: 29: 错误: 程序中有游高的 '\346'
for_asm_open.c: 29: 错误: 程序中有游高的 '\230'
for_asm_open.c: 29: 错误: 程序中有游高的 '\257'
for_asm_open.c: 29: 错误: 程序中有游高的 '\347'
for_asm_open.c: 29: 错误: 程序中有游高的 '\224'
for_asm_open.c: 29: 错误: 程序中有游高的 '\250'
for_asm_open.c: 29: 错误: expected '}' or ';' before 'li'
for_asm_open.c: 29: 错误: 程序中有游高的 '\346'
for_asm_open.c: 29: 错误: 程序中有游高的 '\235'
for_asm_open.c: 29: 错误: 程序中有游高的 '\245'
for_asm_open.c: 29: 错误: 程序中有游高的 '\350'
for_asm_open.c: 29: 错误: 程序中有游高的 '\265'
for_asm_open.c: 29: 错误: 程序中有游高的 '\213'
for_asm_open.c: 29: 错误: 程序中有游高的 '\345'
for_asm_open.c: 29: 错误: 程序中有游高的 '\200'
for_asm_open.c: 29: 错误: 程序中有游高的 '\274'
for_asm_open.c: 29: 错误: 程序中有游高的 '\357'
for_asm_open.c: 29: 错误: 程序中有游高的 '\274'
for_asm_open.c: 29: 错误: 程序中有游高的 '\346'
for_asm_open.c: 29: 错误: 程序中有游高的 '\265'
for_asm_open.c: 29: 错误: 程序中有游高的 '\256'
for_asm_open.c: 29: 错误: 程序中有游高的 '\347'
for_asm_open.c: 29: 错误: 程序中有游高的 '\202'
for_asm_open.c: 29: 错误: 程序中有游高的 '\271'
for_asm_open.c: 29: 错误: 程序中有游高的 '\346'
for_asm_open.c: 29: 错误: 程序中有游高的 '\225'
for_asm_open.c: 29: 错误: 程序中有游高的 '\260'
for_asm_open.c: 29: 错误: 程序中有游高的 '\347'
for_asm_open.c: 29: 错误: 程序中有游高的 '\261'
for_asm_open.c: 29: 错误: 程序中有游高的 '\273'
for_asm_open.c: 29: 错误: 程序中有游高的 '\345'
for_asm_open.c: 29: 错误: 程序中有游高的 '\236'
for_asm_open.c: 29: 错误: 程序中有游高的 '\213'
```

Debug完毕



```
for_asm_open.c
* 编译选项: -std=c99 -O2 -fcommon -fPIE -pie -fstack-protector-strong -fstack-clash-protection -fcf-protection -fdebug-prefix-map=/usr/src/linux-headers-5.10.0-28-generic:/usr/src/linux-headers-5.10.0-28-generic -fdebug-prefix-map=/usr/src/linux-headers-5.10.0-28-generic:/usr/src/linux-headers-5.10.0-28-generic
* 编译时间: 2022-11-10 10:20:35
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{
    float a[200000]; // 定义数组变量
    int i = 0;
    int c, len = 200000;
    int ans;
    int duration;
    clock_t start, finish; // 定义时间变量
    for (i = 0; i < 200000; i++)
    {
        a[i] = 5.0;
    }
    start = clock(); // 开始计时
    int iter = 20;
    for (i = 0; i < iter; i++)
    {
        __asm__(
            "li.s $f2, 10.0\n\t" // 采用指令 li.s 对 f2 进行赋值, 并赋值为 10.0 (浮点型), 在寄存器类型里面
            // 是用 li 来赋值, 浮点数据类型则使用 li.s
            "Loop: \n\t" // 设置循环
            "lwi $f0, 0(%a)\n\t" // 这一句是把值赋给 f0: 这里的 a 用于外部参数输入, 用%a表示; 0(%a)
            // 指的是对 a 这个外部参数偏移 0 个字节
            "lwi $f6, 4(%a)\n\t" // 这一句是把值赋给 f6: 这里的 a 用于外部参数输入, 用%a表示; 4(%a)
            // 指的是对 a 这个外部参数偏移 4 个字节
            "lwi $f10, 8(%a)\n\t" // 这一句是把值赋给 f10: 这里的 a 用于外部参数输入, 用%a表示; 8(%a)
            // 指的是对 a 这个外部参数偏移 8 个字节
            "lwi $f14, 12(%a)\n\t" // 这一句是把值赋给 f14: 这里的 a 用于外部参数输入, 用%a表示; 12(%a)
            // 指的是对 a 这个外部参数偏移 12 个字节
            "add.s $f8, $f0, $f2\n\t" // 这一句是把 f0 和 f2 相加, 并赋值给 f8
            "add.s $f6, $f6, $f2\n\t" // 这一句是把 f6 和 f2 相加, 并赋值给 f6
            "add.s $f12, $f10, $f2\n\t" // 这一句是把 f10 和 f2 相加, 并赋值给 f12
            "add.s $f16, $f14, $f2\n\t" // 这一句是把 f14 和 f2 相加, 并赋值给 f16
            "swi $f8, 0(%a)\n\t" // 保存结果
            "swi $f6, 4(%a)\n\t" // 保存结果
            "swi $f12, 8(%a)\n\t" // 保存结果
            "swi $f16, 12(%a)\n\t" // 保存结果
            "addiu $i, $i, 4\n\t" // 指针偏移 4, 因为我们做了 4 次循环展开, 每个数有 4 个字节, 4*4=16
            "bne $i, $len, Loop\n\t" // 是否是否跳出循环
        );
        // 以这个 ans 没有实际意义, 输出, 这是 asm 编译需要的, 这句话不能删掉, 但此程序不需要输出, 所
        : [a] "=r" (a), [len] "=r" (len) // 外部输入, 把数组和数组长度传入
    );
    finish = clock(); // 结束计时
    duration = (int)(finish - start) / iter;
    printf("the clock time is: %d\n", duration);
}
```

运行

./for ./ forasm ./ forasmopen

```
for_asm_open.c: 51: 错误: 程序中有游高的 '\271'
for_asm_open.c: 51: 错误: 程序中有游高的 '\211'
[loongson@localhost 桌面]$ ge
gedit          gendiff          genisoimage      getconf          getfacl          getop
gegl          genhomedircon  geqn             getenforce       getkey           getop
gencat        genhostid      getcap           getent           getkeycodes      getpc
[loongson@localhost 桌面]$ ge
gedit          gendiff          genisoimage      getconf          getfacl          getop
gegl          genhomedircon  geqn             getenforce       getkey           getop
gencat        genhostid      getcap           getent           getkeycodes      getpc
[loongson@localhost 桌面]$ ge
gedit          gendiff          genisoimage      getconf          getfacl          getop
gegl          genhomedircon  geqn             getenforce       getkey           getop
gencat        genhostid      getcap           getent           getkeycodes      getpc
[loongson@localhost 桌面]$ gedit for_asm_open.c
[loongson@localhost 桌面]$ vim for_asm_open.c
[loongson@localhost 桌面]$ gcc for_asm_open.c -o forasmopen
[loongson@localhost 桌面]$ ./for ./forasm ./for
forasmopen      for_asm_open.c  for_asm_s.c    f
[loongson@localhost 桌面]$ ./for ./forasm ./forasmopen
the clock time is: 20000000
[loongson@localhost 桌面]$ ./for
I
the clock time is: 20000000
[loongson@localhost 桌面]$ ./forasm
bash: ./: is a directory
[loongson@localhost 桌面]$ ./forasm
the clock time is: 20000000
[loongson@localhost 桌面]$ ./forasmopen
the clock time is: 6000
[loongson@localhost 桌面]$
```


四、 实验过程分析与讨论

发现./for ./ forasm的执行时间相同，未有太大区别，但感觉后者应该相对快一点

而 ./ forasmopen的执行时间缩小了4的数量级，大大缩短了时间，证明并行操作确实取得了很大的功效。

五、指导教师意见

指导教师签字：卢洋