

**实验报告**



**题目： Linux环境和GCC工具链**

**班 级： 2023211301**

**学 号： 2023210896**

**姓 名： 王书翰**

**学 院： 计算机学院（国家示范性软件学院）**

**2024年10 月15日**

一、实验目的

1. 熟悉linux系统的常用命令；
2. 掌握gcc编译器的使用方法；
3. 掌握gdb的调试工具使用；
4. 掌握objdump反汇编工具使用；
5. 理解反汇编程序（对照源程序与objdump生成的汇编程序）。
6. 实验环境+-

列举所使用的软件工具

1. Windows powershell 1.20.11781.0
2. Vim 8.2.2121
3. gdb 12.1

三、实验内容

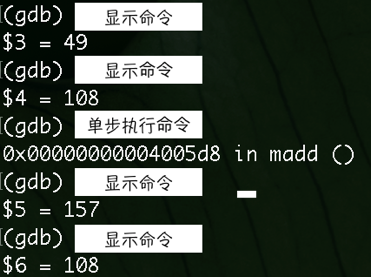
现有两个int型数组a[i]=i-50，b[i]=i+y，其中y取自于学生本人学号2022211x\*y的个位。登录bupt1服务器，在linux环境下使用vi编辑器编写C语言源程序，完成数组a ← a+b的功能，规定数组长度为100，函数名为madd（），数组a，b均定义在函数内，使用gcc编译该程序（使用-g -no-pie -fno-stack-protector选项），

1. 使用objdump工具生成汇编程序，找到madd函数的汇编程序，给出截图；
2. 用gdb进行调试，练习下列gdb命令，给出截图；

gdb、kill、quit、break（break \*0x地址 if $rax==401234）、delete、clear、info break、run、continue、nexti、stepi、disassemble、list、print、x、info reg、watch

1. 找到a[i]+b[i]对应的汇编指令，指出a[i]和b[i]位于哪个寄存器中，给出截图；
2. 使用单步执行命令及gdb相关命令，显示a[xy]+b[xy]对应的汇编指令执行前后操作数寄存器十进制和十六进制的值，其中x，y取自于学生本人学号2022211x\*y的百位和个位。

学号2022211999，a[99]+b[99]单步执行前后的参考截图如下（实际命令未显示出）：



四、实验步骤及实验分析

**1、启动powershell登录bupt1服务器**

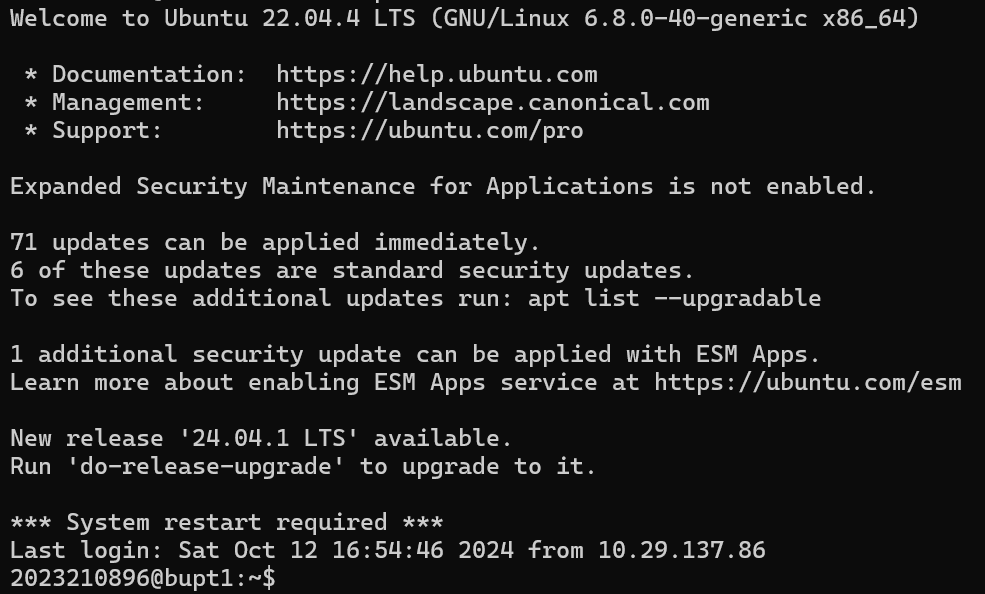


图 1

**2、启动vi编辑器新建lab1.c文件编写主函数与madd()函数**

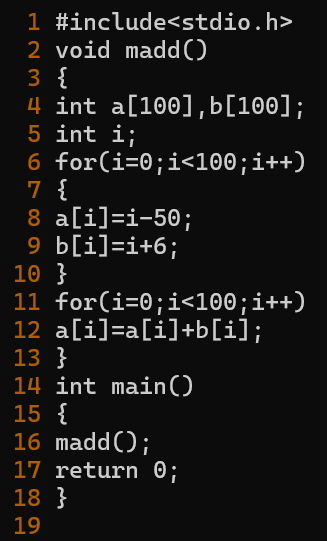


图 2

**3、使用gcc编译该程序**

使用gcc -g -no-pie -fno-stack-protector lab1.c -o lab1指令编译lab1.c文件，后使用ls指令查看文件列表发现存在两个文件：lab1.c与lab1，证明程序成功编译

**4、使用objdump工具生成汇编程序，找到madd函数的汇编程序**

使用objdump -d -S lab1 > lab1.s 指令生成汇编程序并命名为lab1.s，后使用cat lab1.s指令打开lab1.s文件找到madd函数并截图（如图3所示）

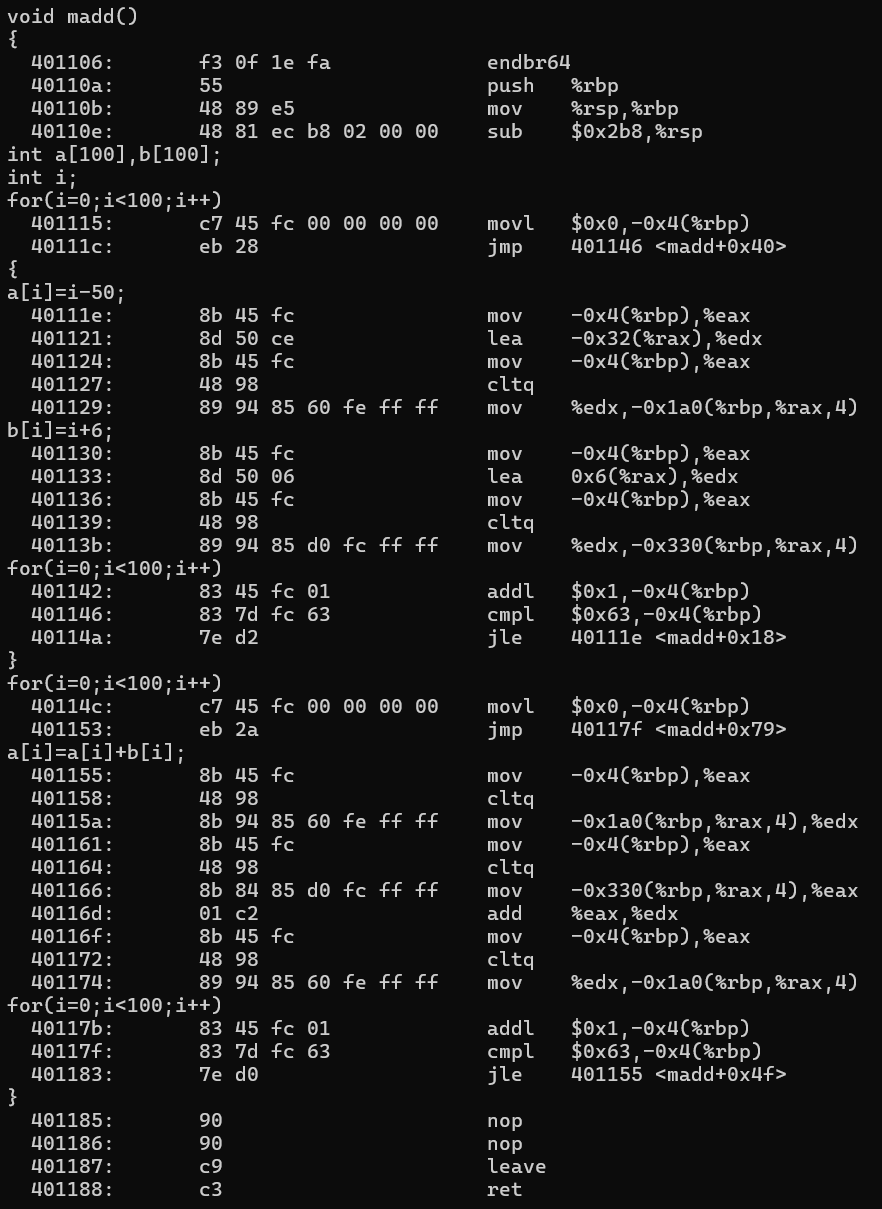


图 3

**5、使用gdb调试并实现各项功能**

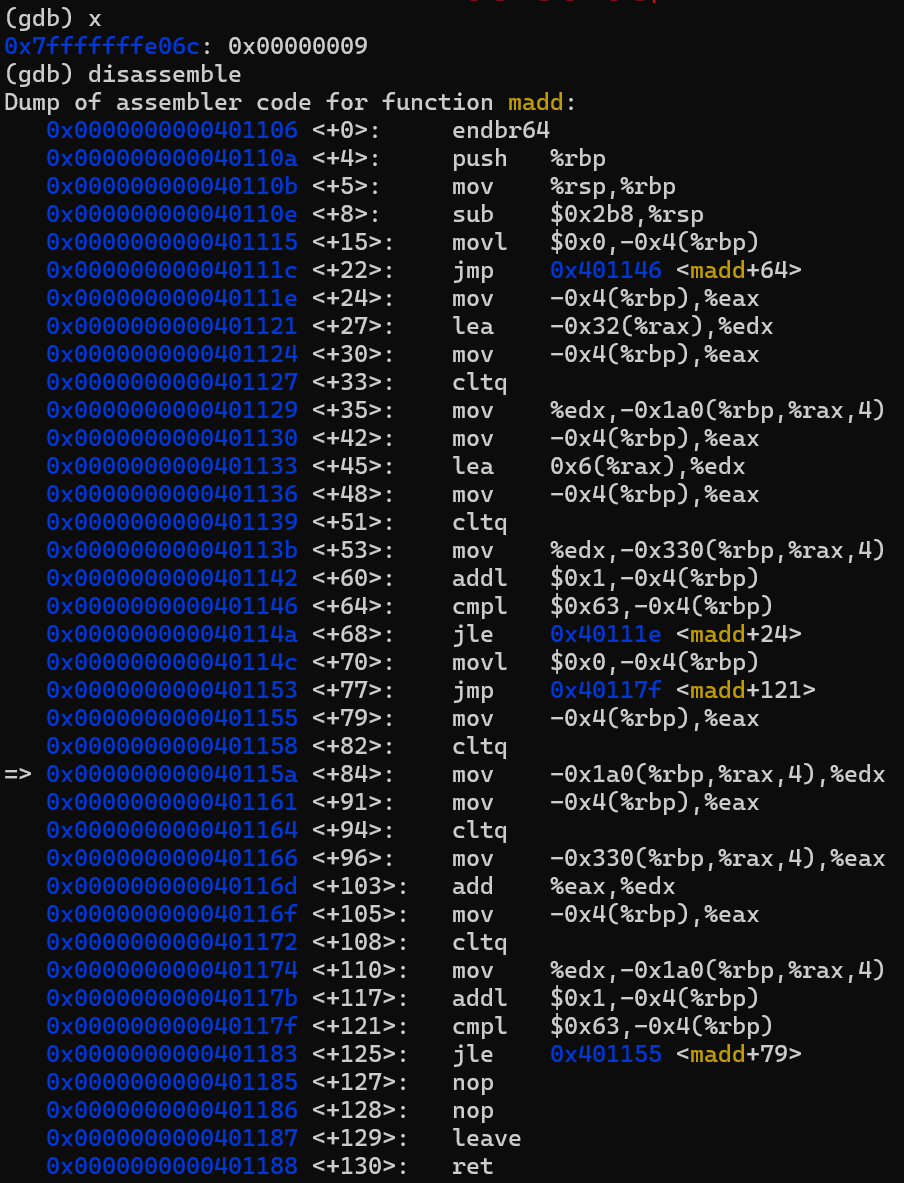


图 9 x, disassemble

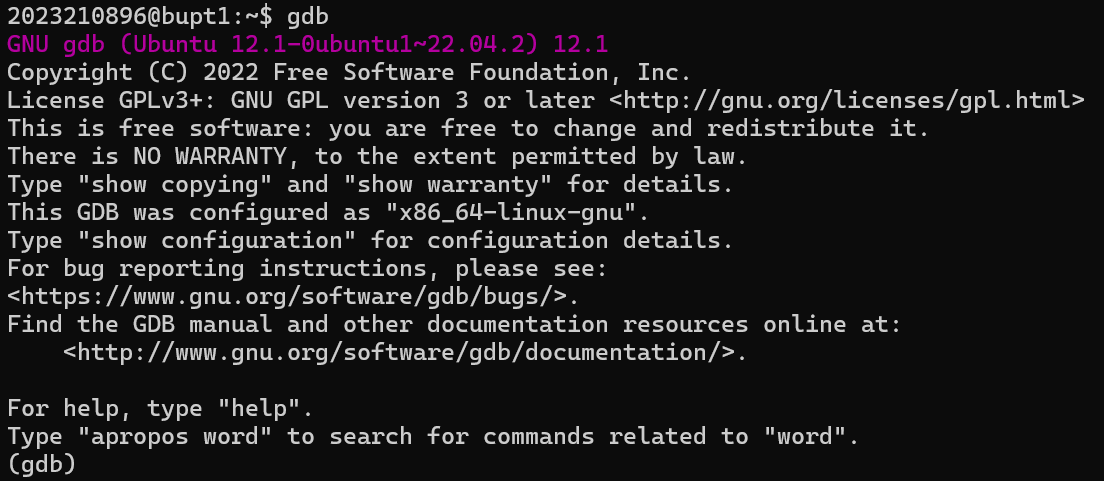


图 4 gdb

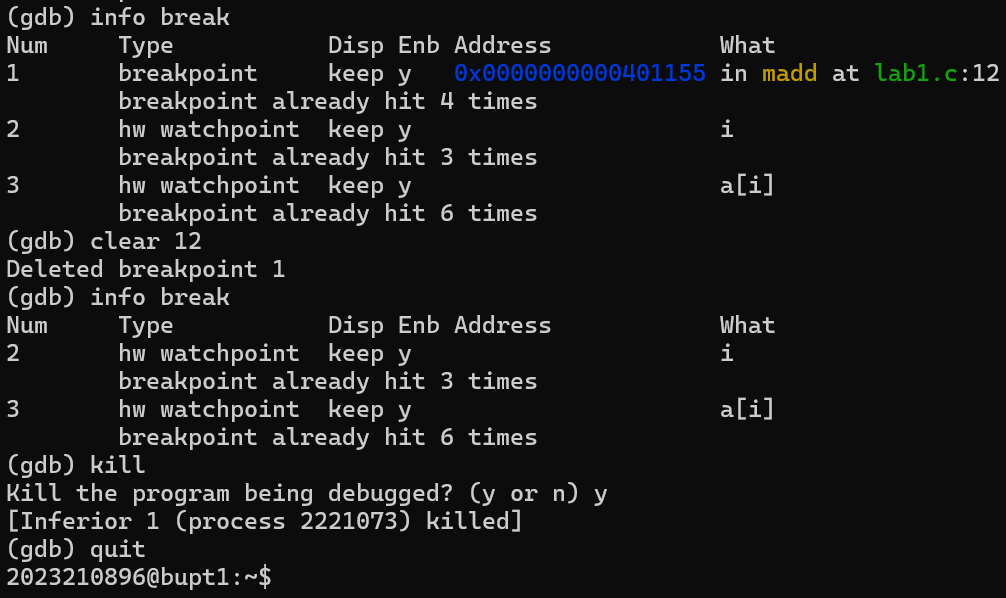


图 5 kill, quit, clear

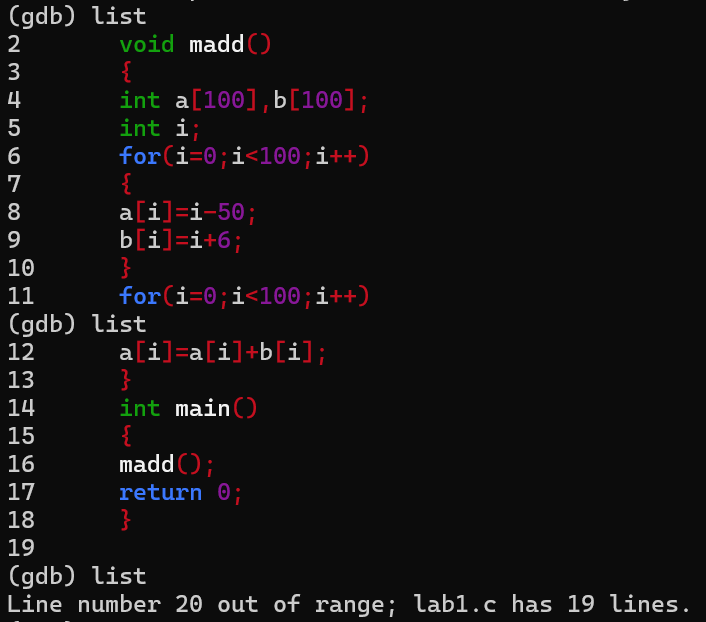


图 10 list

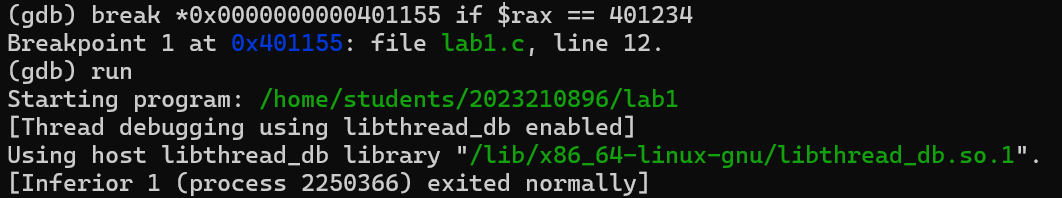


图 6 break

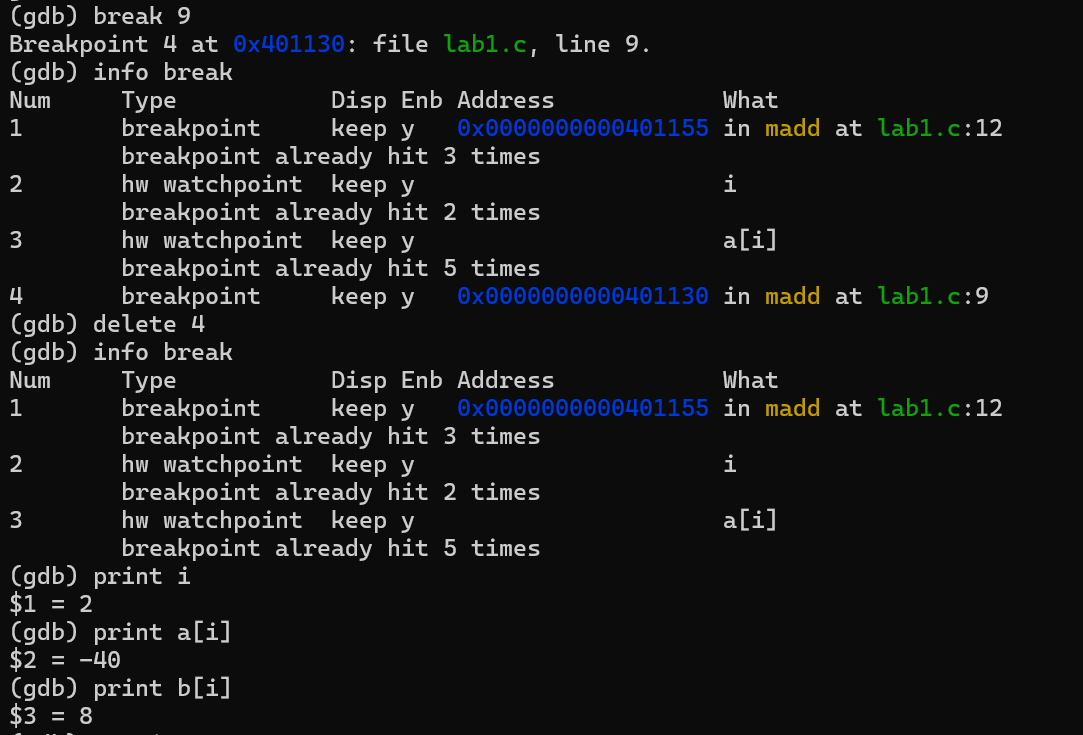


图 7 info break, delete, print

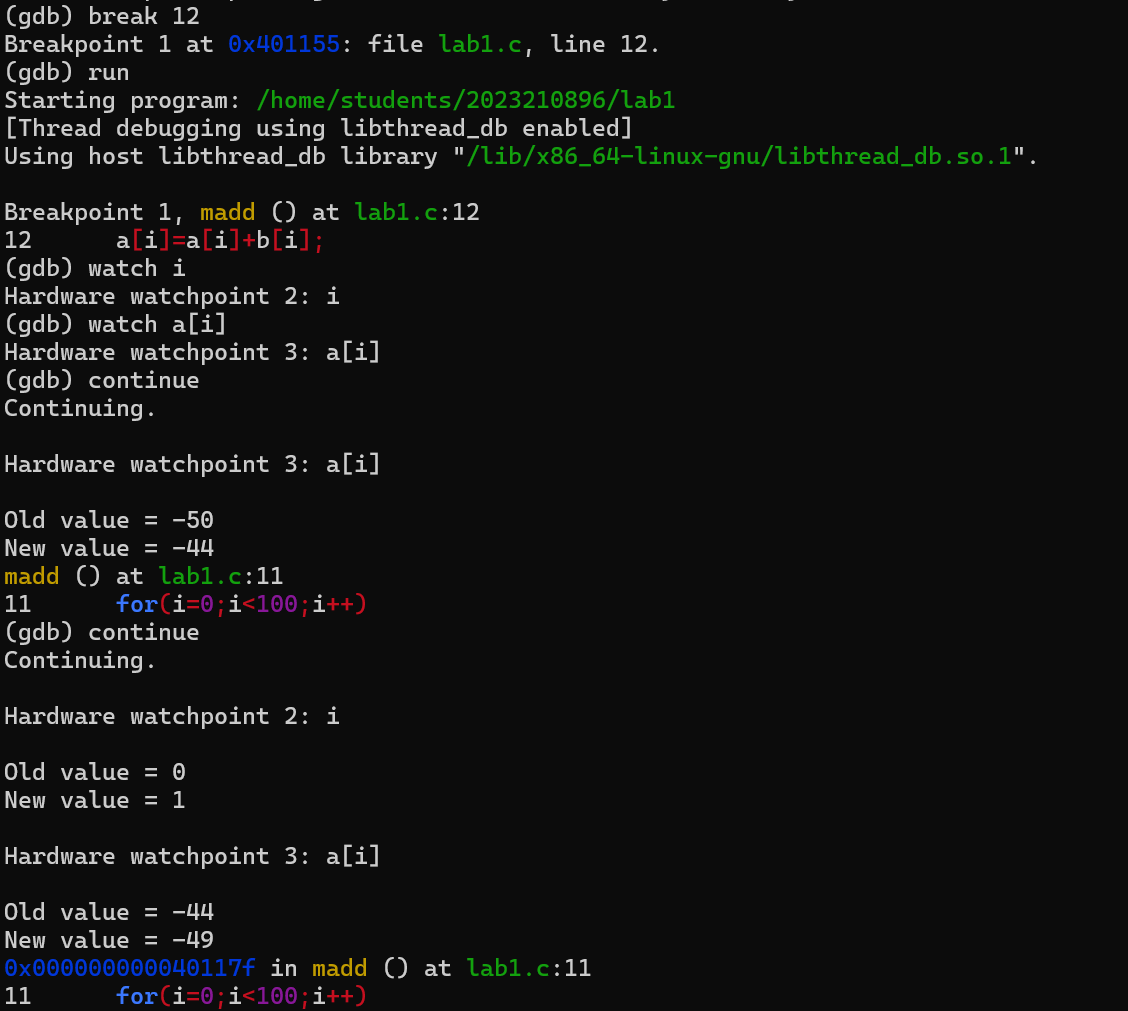


图 13 run, continue, watch

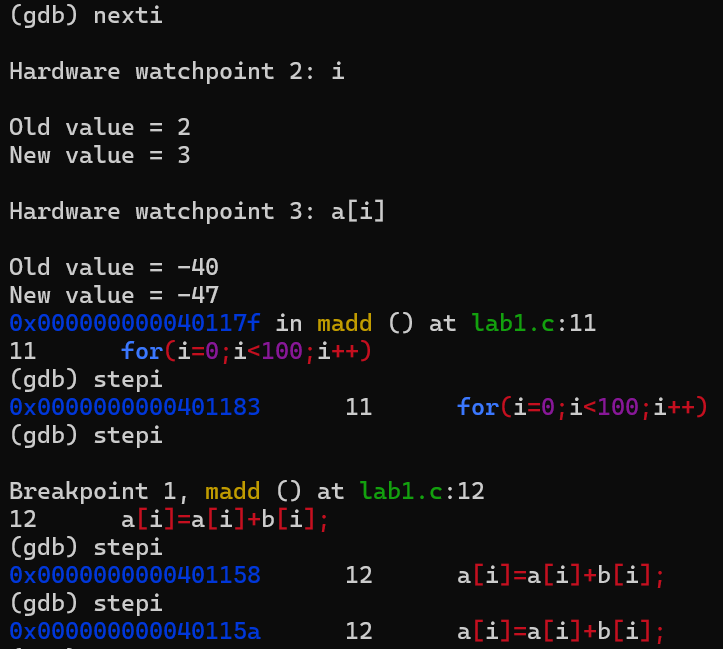


图 11 nexti, stepi

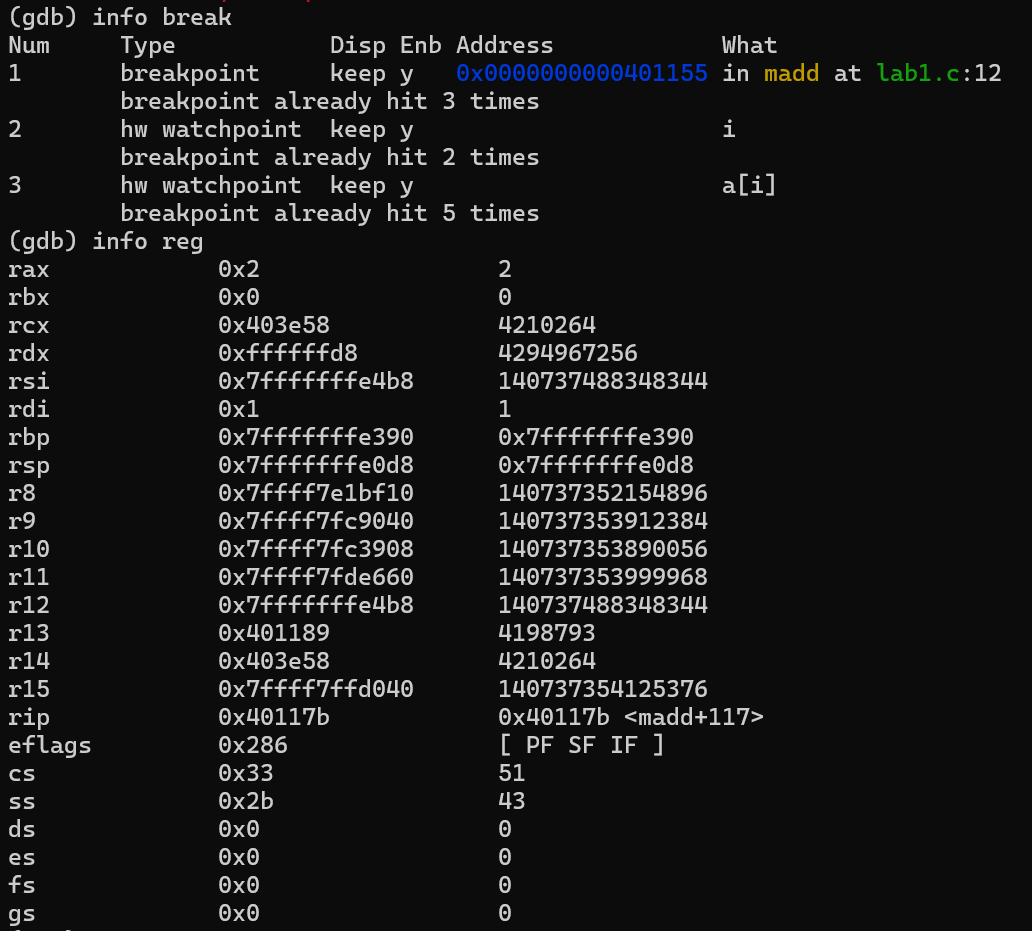


图 12 info reg

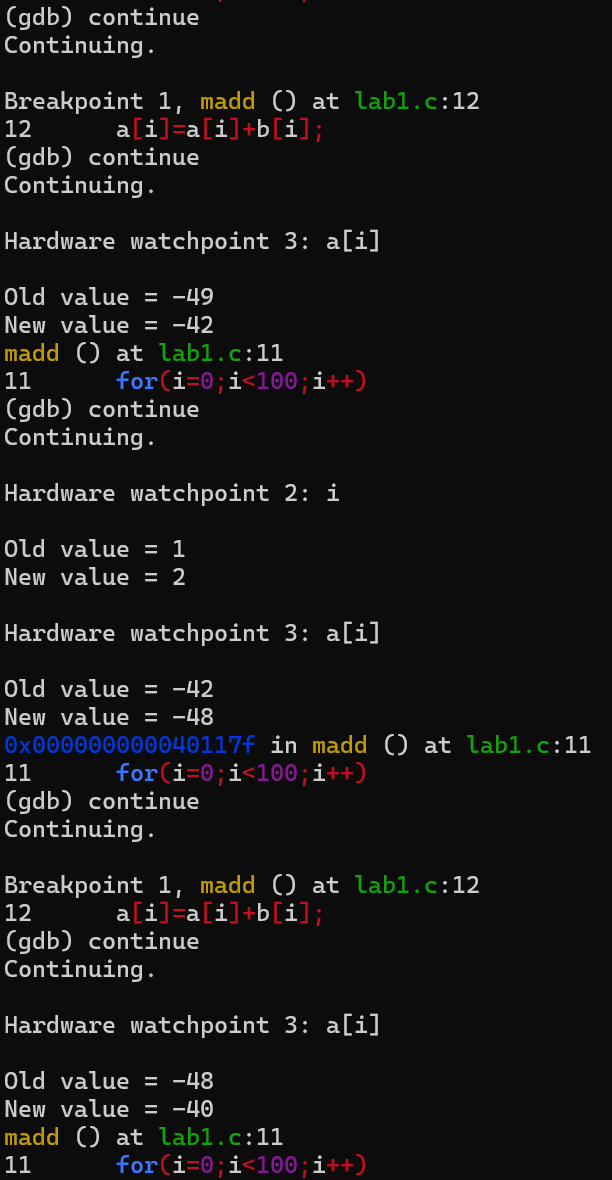


图 13 run, continue, watch

**6、找到a[i]+b[i]对应的汇编指令，指出a[i]和b[i]位于哪个寄存器中**

在a[i]=a[i]+b[i]行设置断点，运行到断点前用disas指令查看程序执行到哪一行汇编指令，之后重复用si指令运行下一行汇编指令直到a[i]+a[i]+b[i]运行结束，最后再次用disas指令查看程序运行到哪一行汇编指令，第一次和第二次显示的汇编指令区间就是a[i]=a[i]+b[i]所对应的汇编指令

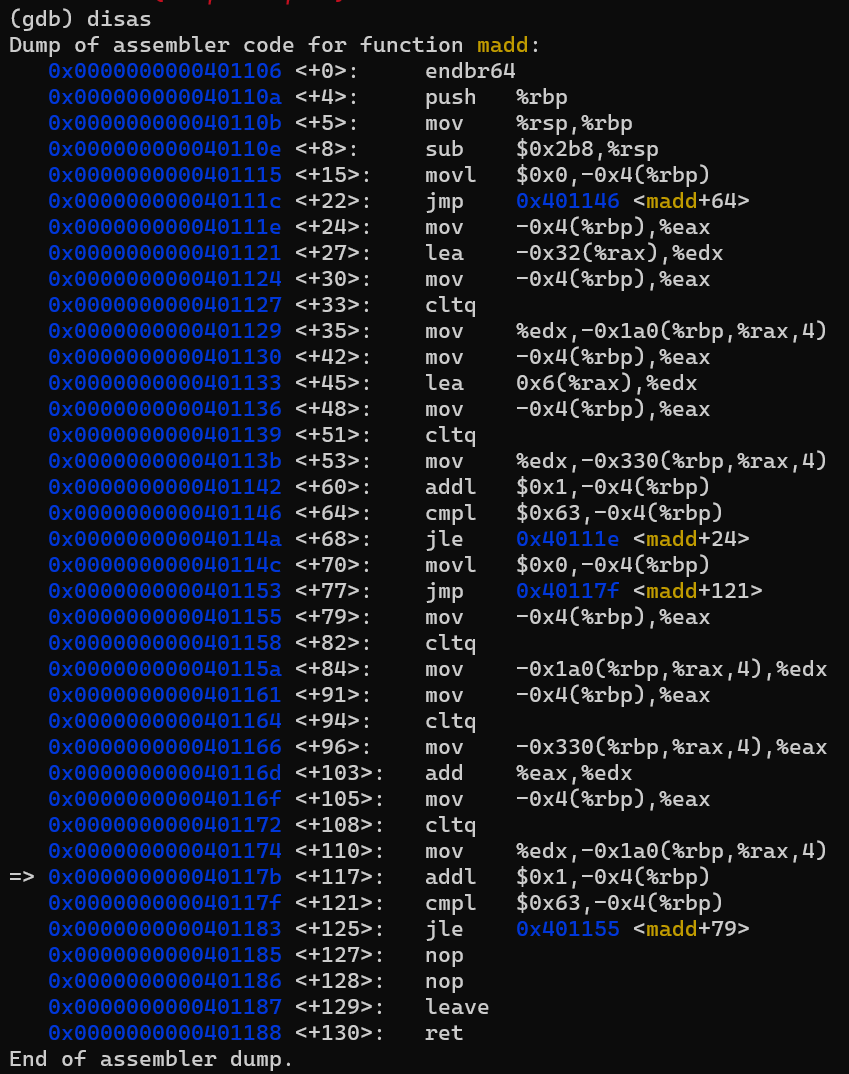


图 15 a[i]=a[i]+b[i] 运行后

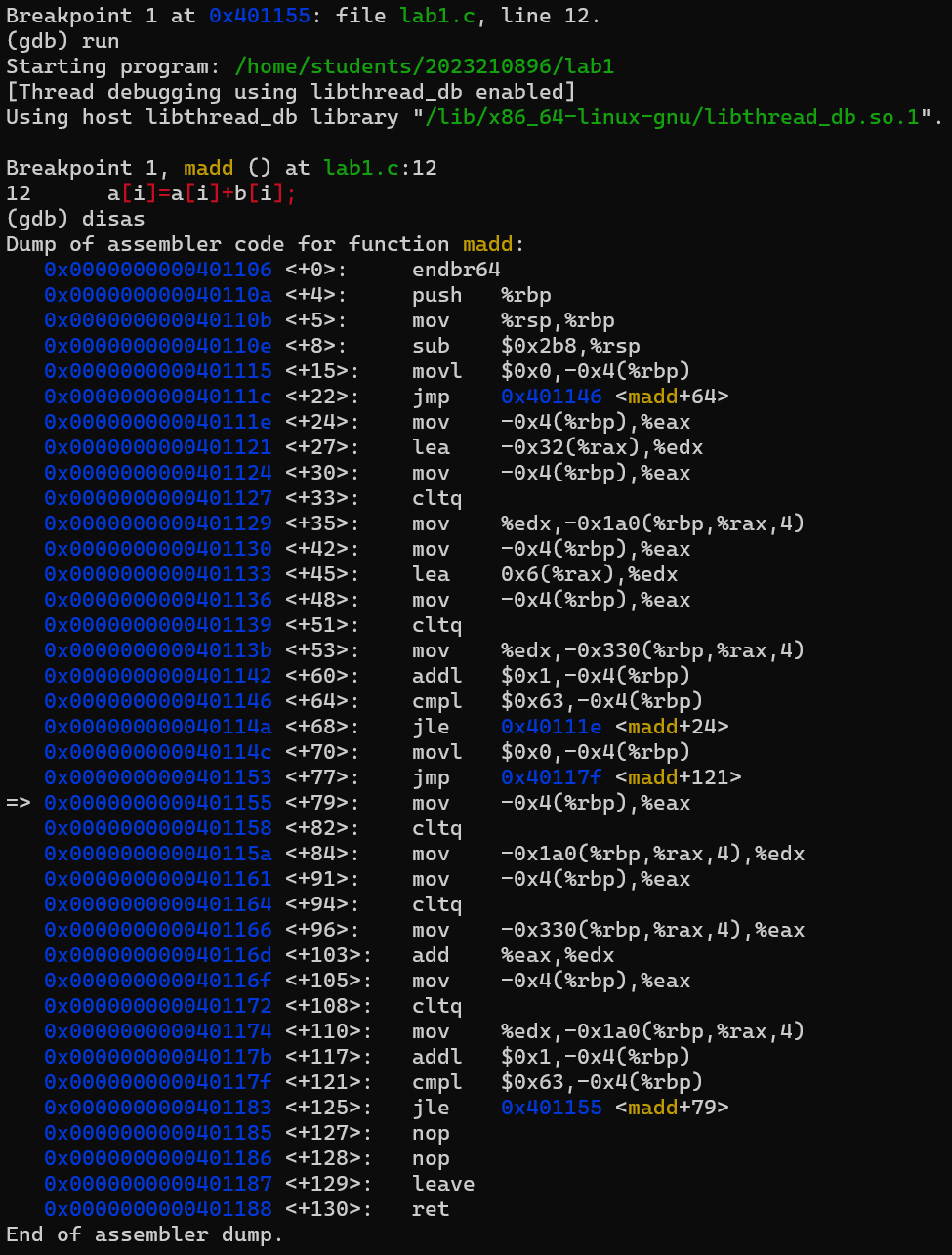


图 14 a[i]=a[i]+b[i] 运行前

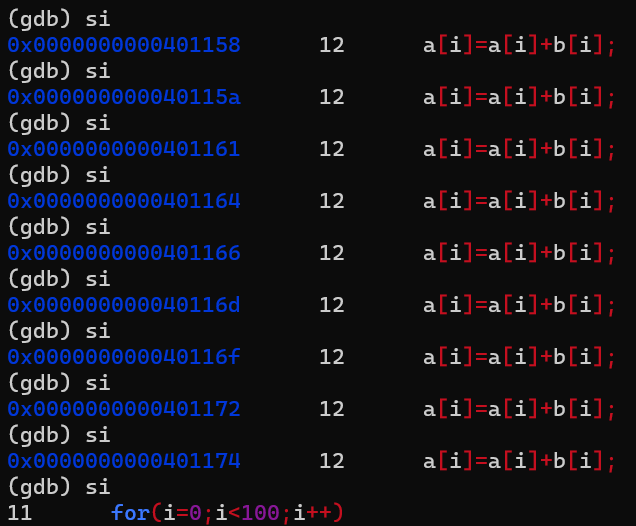


图 16 a[i]=a[i]+b[i] 运行中

由图14，15，16可知运行前停在<+79>处，经过10次si指令后停在<+117>处，故a[i]=a[i]+b[i]对应的汇编指令就是<+79>处到<+117>处。

继续分析<+79>处到<+117>处汇编指令，发现执行过程中%eax与%edx寄存器储存的值发生了改变，故推测其与a[i]，b[i]有关

为知道具体a[i]与b[i]分别储存在哪个寄存器中，分别使用指令watch a[i]，b[i]，$eax，$edx的值，在逐次使用si指令观察此四值发生的变化。

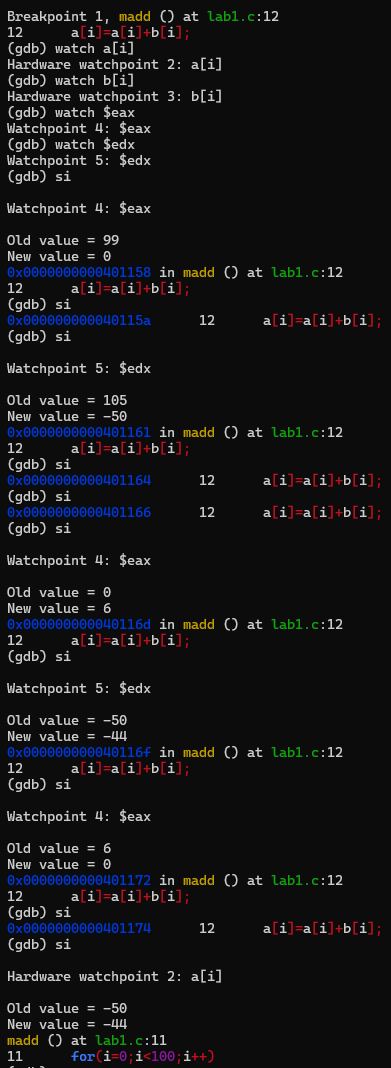
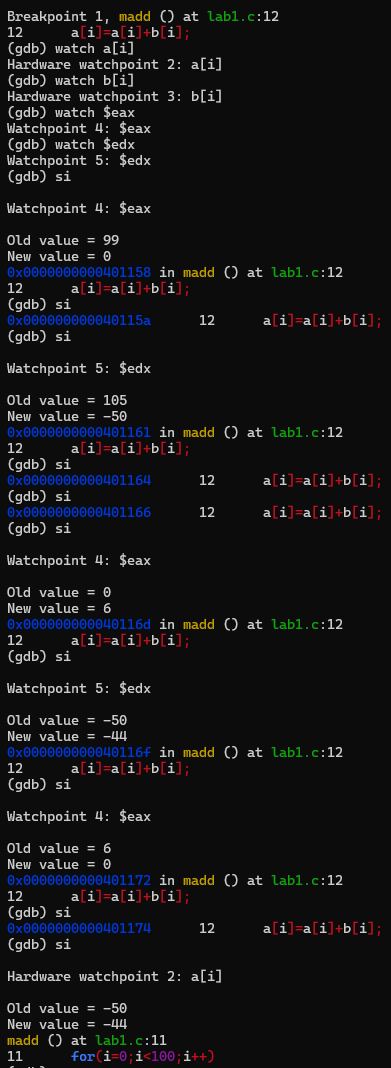


图 17

图 18

由运行结果可知a[i]与%edx的变化相同，故a[i]储存在%edx中。又发现%eax原本的值与i相同，但在执行过程中变成了b[i]的值，在运行结束后由恢复了i的值，故b[i]储存在%eax中，同时%eax也储存着i的值

**7、使用单步执行命令及gdb相关命令，显示a[xy]+b[xy]对应的汇编指令执行前后操作数寄存器十进制和十六进制的值，其中x，y取自于学生本人学号2022211x\*y的百位和个位。**

本人学号为2023210896，故x=8，y=6，要求显示a[86]=a[86]+b[86]前后操作数寄存器的值。故设置i==63时的断点，先打印两个寄存器的十进制和十六进制的值，随后用si指令逐步运行到a[i]=a[i]+b[i]指令结束，再次打印两个寄存器的十进制和十六进制的值

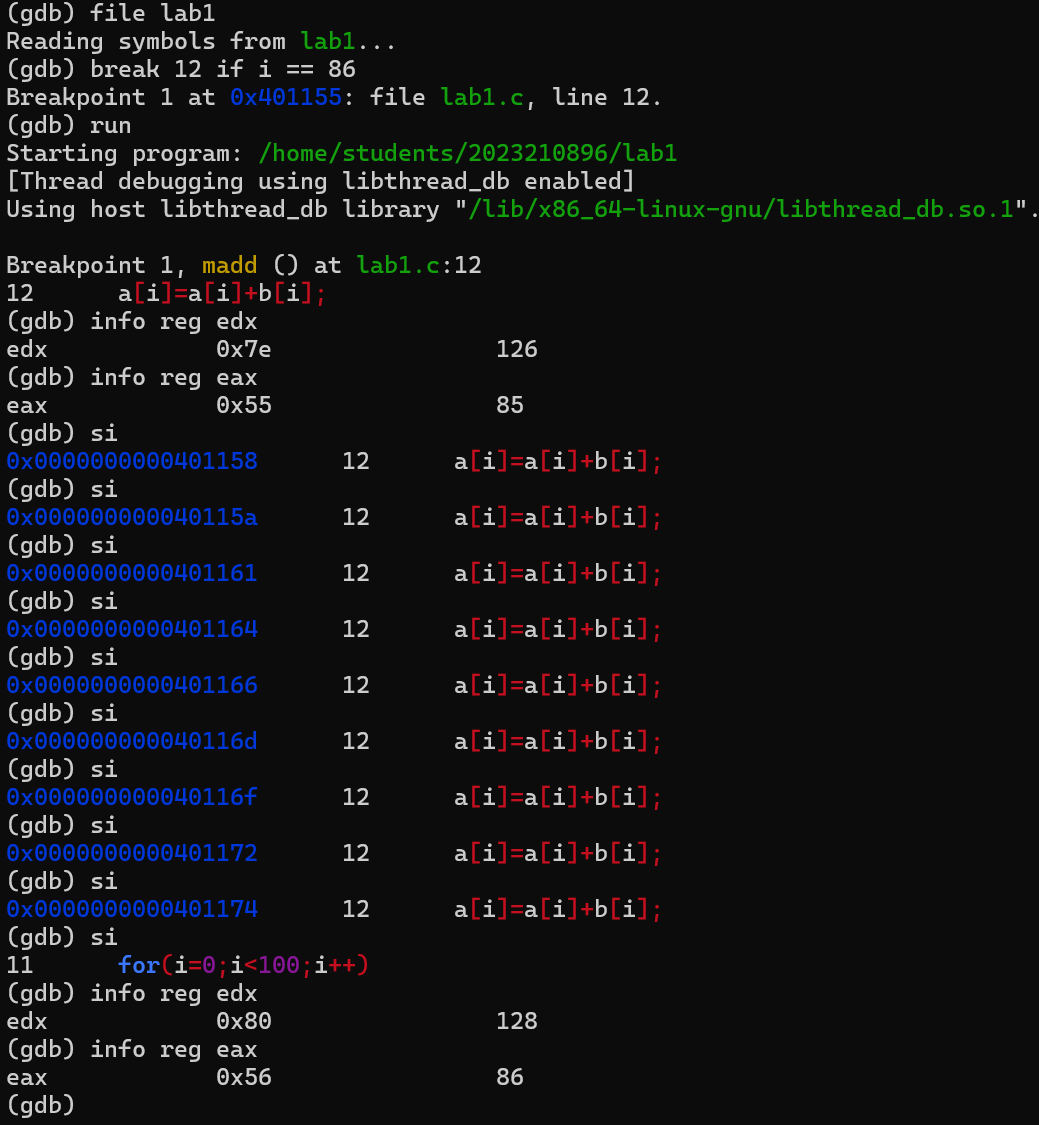


图 19

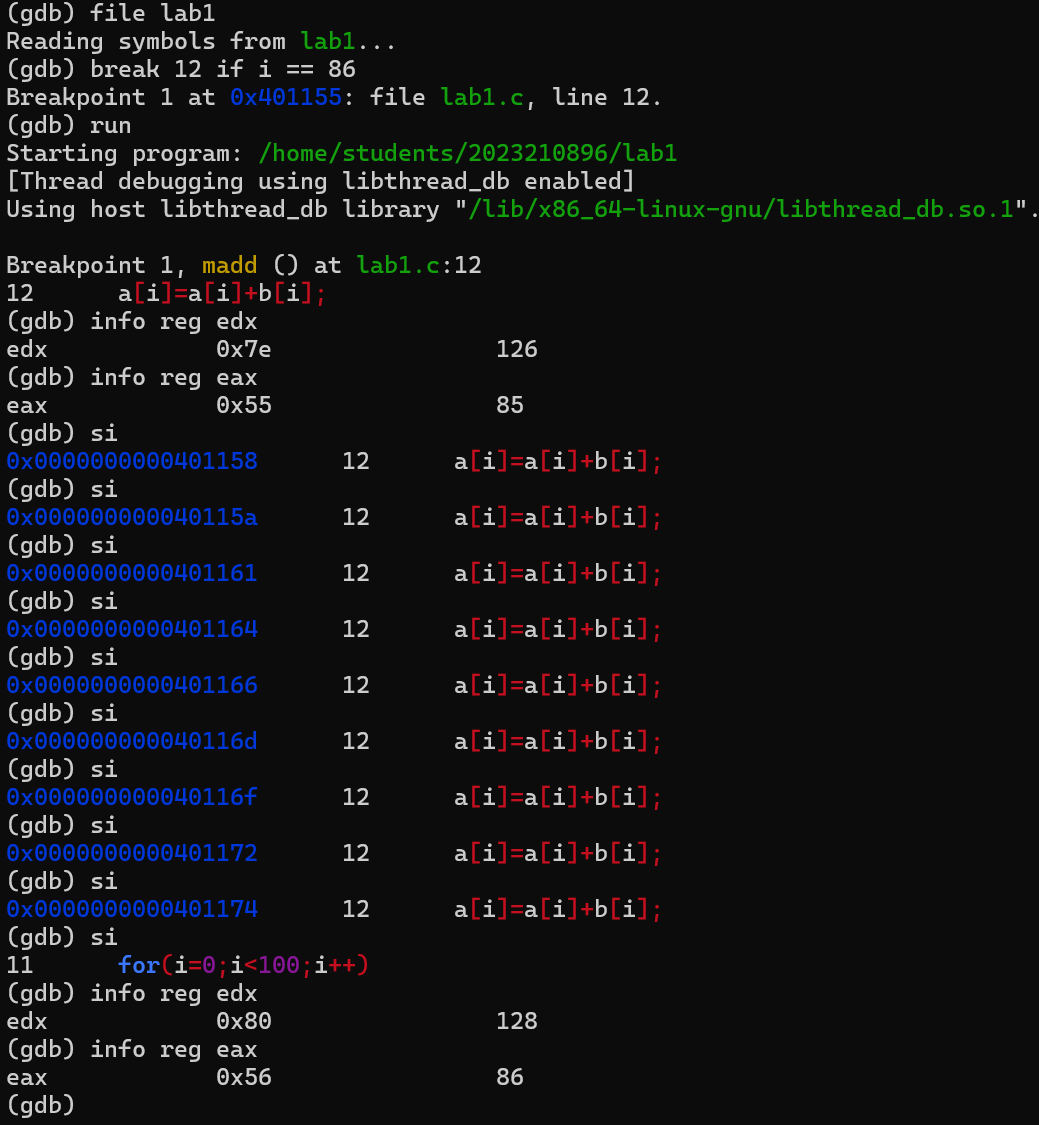


图 20

由运行截图可得a[86]=a[86]+b[86]执行前edx=126/0x7e，eax=85/0x55；执行后edx=128/0x80，eax=86/0x56

五、总结体会

总结心得（包括实验过程中遇到的问题、如何解决的、意见和建议等）

1、实验内容总结：这次实验我掌握了vi编辑器的基本功能与gdb调试工具的基本用法。同时我将课堂上所学的课本内容运用到实践中，这解决了我在实验中遇到的各种问题。

2、实验过程中遇到的问题：首先是在使用vi编辑器时感到不适应，相比集成开发环境vi编辑器更难以操作。其次是在linux环境中通过输入各种不同的指令来实现不同的功能，这让平时习惯于点击执行指令的我难以熟练运用各种指令。

3、解决的方法：通过仔细阅读实验指南与网上的各种教程，我渐渐熟悉了vi编辑器的各项操作。同时我逐个使用linux指令和gdb调试指令，在实验中逐渐认识到这样指令操作的优越性与准确性。

4、意见与建议：我建议同学们在实验完成后将自己的问题与收获同其他同学分享，让大家能解决共性的问题同时巩固相关知识，掌握在解决各种各样的问题中学习提高的能力。