实验三 Linux基础编程

一、实验目的

通过本次实验使学生深入理解Linux操作系统中进行C/C++语言开发的过程。熟练掌握在Linux操作系统命令模式下，使用VI进行C/C++语言源程序的编写、使用GCC/G++编译、运行及使用GDB调试程序。

二、实验原理

Linux基础编程包括在Linux操作系统下的C/C++语言源程序的编译、运行、及调试。本次实验学生需要重点掌握的是GCC编译器以及GDB调试工具。

1. GCC编译器

GCC是GNU C Compiler的缩写，它是GNU/Linux下最好的编译器之一。此编译器性能稳定，功能强大，而且文档齐全，大部分的自由软件都是用它进行编译的。如果开发者使用C或C++语言，那么可以优先考虑使用GCC编译器。

GCC编译器能将C/C++语言源程序编译成目标程序，然后将目标程序连接成可执行文件，如果没有指定可执行文件的名字，GCC将自动生成一个叫a.out的可执行程序。

GCC的编译分为以下4个步骤：

1. 预处理：GCC调用cpp程序进行预处理，即分析如#include、#define之类的命令。
2. 编译：GCC调用ccl程序进行编译，它将根据源代码生成汇编语言。
3. 汇编：GCC调用as程序将上一步的结果生成.o目标文件。
4. 连接：GCC调用ld程序将目标文件进行连接，最后生成可执行文件。

GCC的基本用法如下：

GCC [options] [filename]

其中常用的参数选项有：

-o file 用来指定输出的文件名（默认值是a.out）。

-g 同GDB一起使用，产生排错信息。

-c 只进行编译和汇编（不进行连接）。

-S 编译（不汇编或连接）。

-E 只进行预处理。

例：#gcc -g -o sum sum.c

注意：当源程序为C++语言时，使用g++命令，具体用法同上。

1. GDB调试器

Linux包含了一个叫GDB的GNU调试程序。GDB是一个用来调试C和C++程序的调试器，它有非常好的调试特性，能在程序运行时观察程序的内部结构、变量的值以及内存的使用情况等。GDB所提供的常见功能如下：

1. 堆栈跟踪。
2. 监视程序中变量的值。
3. 设置断点使程序在指定的代码行上停止执行。
4. 单步执行代码。

GDB程序调试的对象是可执行文件，如果要让GCC产生的可执行文件可以用来调试，需要在执行GCC命令编译程序时，加上-g参数，例如：

#gcc -g -o sum sum.c

#gdb sum

接下来即可进入GDB调试界面进行调试，如图1所示。其中常用的GDB调试命令有：

list 列出产生执行文件的源代码的一部分。

print 显示变量或表达式的值。

set 修改变量的值。

next 单步执行下一行程序。

run 执行当前被调试的程序。

break 在代码中设置断点，程序执行至此时将被挂起。

clear 清除断点。

continue 从断点处继续执行。

kill 终止正在调试的程序。

quit 终止退出GDB。

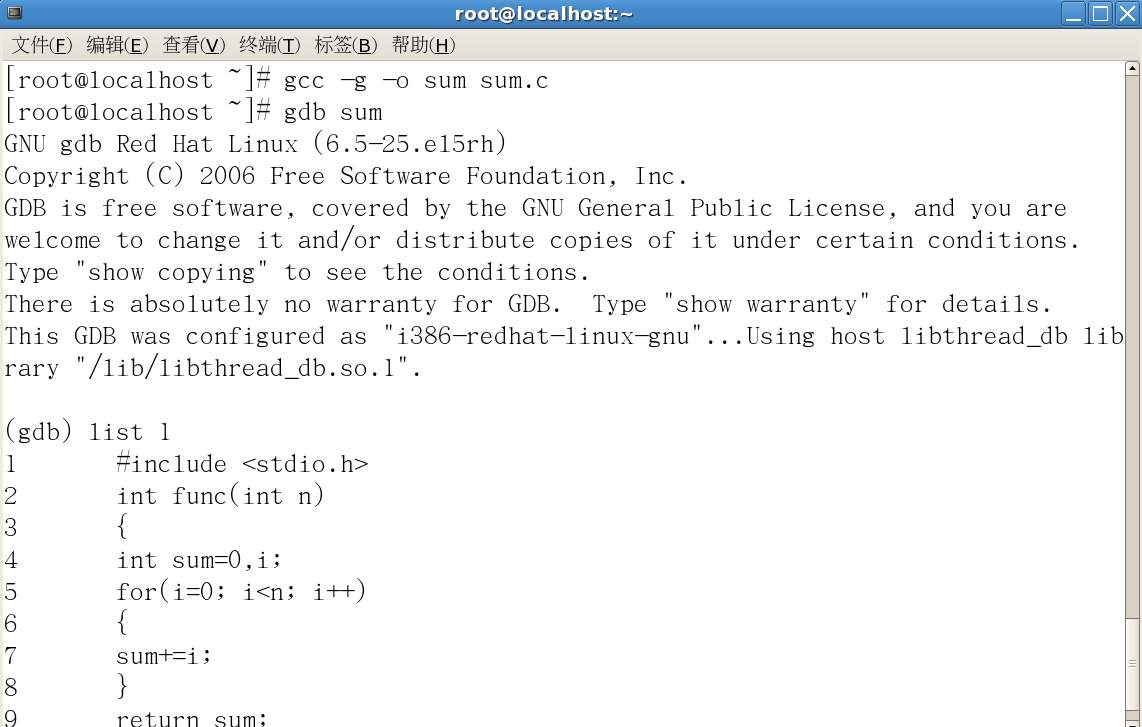


图1 GDB调试示例

三、实验内容

1. **Linux下C/C++语言源程序的编写及编译**

在Linux的命令模式下，掌握C/C++语言的编写、编译、运行：

1）使用VI编写C语言源程序代码如下：

#include <stdio.h>

int func(int n)

{

int sum=0,i;

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

{

sum+=i;

}

return sum;

}

int main()

{

int i;

long result = 0;

for(i=1; i<=100; i++)

{

result += i;

}

printf("result[1-100] = %d \n", result );

printf("result[1-250] = %d \n", func(250) );

}

2）使用GCC进行编译并运行上面的C语言程序。

3）将1）中C语言源程序改写成C++语言源程序，并使用g++进行编译运行。

1. **使用GDB调试程序**

使用GDB对上面所编写的C/C++程序进行调试，具体包含以下操作：

1. 列出源程序代码。
2. 设置断点。
3. 运行程序及单步运行程序。
4. 输出变量当前的值。
5. 中断调试。

四、实验要求

1. 完成实验内容中的1、2的练习，并将运行结果截图保存在实验报告中。

2. 写出在实验过程中所遇见的问题，分析错误原因，及你的解决办法。