

Отчёт по лабораторной работе №9

Архитектура компьютера

Ирина Васильевна Панявкина

Содержание

1 Цель работы

Приобретение навыков написания программ с использованием подпрограмм. Знакомство с методами отладки при помощи GDB и его основными возможностями.

2 Задание

1. Реализация подпрограмм в NASM
2. Отладка программ с помощью GDB
3. Самостоятельное выполнение заданий по материалам лабораторной работы

3 Теоретическое введение

Отладка — это процесс поиска и исправления ошибок в программе. В общем случае его можно разделить на четыре этапа: • обнаружение ошибки; • поиск её местонахождения; • определение причины ошибки; • исправление ошибки.

Можно выделить следующие типы ошибок: • синтаксические ошибки — обнаруживаются во время трансляции исходного кода и вызваны нарушением ожидаемой формы или структуры языка; • семантические ошибки — являются логическими и приводят к тому, что программа запускается, отработывает, но не даёт желаемого результата; • ошибки в процессе выполнения — не обнаруживаются при трансляции и вызывают прерывание выполнения программы (например, это ошибки, связанные с переполнением или делением на ноль). Второй этап — поиск местонахождения ошибки. Некоторые ошибки обнаружить довольно трудно. Лучший способ найти место в программе, где находится ошибка, это разбить программу на части и произвести их отладку отдельно друг от друга. Третий этап — выяснение

причины ошибки. После определения местонахождения ошибки обычно проще определить причину неправильной работы программы. Последний этап — исправление ошибки. После этого при повторном запуске программы, может обнаружиться следующая ошибка, и процесс отладки начнётся заново.

4 Выполнение лабораторной работы

Реализация подпрограмм в NASM

Создаю каталог для программ лабораторной работы №9, а также файл lab9-1.asm и копирую в текущий каталог файл in_out.asm с помощью утилиты cp, т.к. он будет использоваться во время выполнения самостоятельной работы (рис. 1).

```
irina_panyavkina@vbox:~$ mkdir ~/work/arch-pc/lab09
irina_panyavkina@vbox:~$ cd ~/work/arch-pc/lab09
irina_panyavkina@vbox:~/work/arch-pc/lab09$ touch lab9-1.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ mousepad lab9-1.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ cp ~/Downloads/in_out.asm in_out.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ls
in_out.asm  lab9-1.asm
```

Рис. 1: Создание каталога и файла для программы, создание копии внешнего файла

Открываю созданный файл и вставляю в него скопированную программу из листинга, компилирую и запускаю его, программа выполняет вычисление функции (рис. 2).

```
irina_panyavkina@vbox:~/work/arch-pc/lab09$ mousepad lab9-1.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ nasm -f elf lab9-1.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-1 lab9-1.o
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ./lab9-1
Введите x: 7
2x+7=21
irina_panyavkina@vbox:~/work/arch-pc/lab09$
```

Рис. 2: Запуск программы из листинга

Изменяю программу, добавив в неё подпрограмму, таким образом, чтобы она вычисляла значение функции для выражения $f(g(x))$ (рис. 3).

```
irina_panyavkina@vbox:~/work/arch-pc/lab09$ mousepad lab9-1.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ nasm -f elf lab9-1.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-1 lab9-1.o
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ./lab9-1
Введите x: 7
2(3x-1)+7=47
```

Рис. 3: Изменение программы первого листинга

Код программы:

```
%include 'in_out.asm'
```

```
SECTION .data
```

```
msg: DB 'Введите x: ', 0
```

```
result: DB '2(3x-1)+7=', 0
```

```
SECTION .bss
```

```
x: RESB 80
```

```
res: RESB 80
```

```
SECTION .text
```

```
GLOBAL _start
```

```
_start:
```

```
mov eax, msg
```

```
call sprint
```

```
mov ecx, x
```

```
mov edx, 80
```

```
call sread
```

```
mov eax, x
```

```
call atoi
```

```
call _calcul
```

```
mov eax, result
```

```
call sprint
```

```
mov eax, [res]
```

```
call iprintLF
```

```
call quit
```

```
_calcul:
```

```
push eax
```

```
call _subcalcul
```

```
mov ebx, 2
```

```
mul ebx
```

```
add eax, 7
```

```
mov [res], eax
```

```
pop eax
```

```
ret
```

```
_subcalcul:
```

```
mov ebx, 3
```

```
mul ebx
```

```
sub eax, 1
```

```
ret
```

Отладка программ с помощью GDB

В созданный файл копирую программу второго листинга, транслирую с созданием файла листинга и отладки, компоную и запускаю в отладчике (рис. 4).

```
irina_panyavkina@vbox:~/work/arch-pc/lab09$ nasm -f elf -g -l lab9-2.lst lab9-2.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-2 lab9-2.o
irina_panyavkina@vbox:~/work/arch-pc/lab09$ gdb lab9-2
GNU gdb (Fedora Linux) 14.2-1.fc40
Copyright (C) 2023 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab9-2...
(gdb) □
```

Рис. 4: Запуск программы в отладчике

Запустив программу командой run, я убедилась в том, что она работает корректно (рис. 5).

```
irina_panyavkina@fedora:~/work/arch-pc/lab09

irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/study/2024... x

irina_panyavkina@vbox:~/work/arch-pc/lab09$ nasm -f elf -g -l lab9-2.lst lab9-2.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-2 lab9-2.o
irina_panyavkina@vbox:~/work/arch-pc/lab09$ gdb lab9-2
GNU gdb (Fedora Linux) 14.2-1.fc40
Copyright (C) 2023 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab9-2...
(gdb) run
Starting program: /home/irina_panyavkina/work/arch-pc/lab09/lab9-2

This GDB supports auto-downloading debuginfo from the following URLs:
<https://debuginfod.fedoraproject.org/>
Enable debuginfod for this session? (y or [n]) y
Debuginfod has been enabled.
To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
Hello, World!
[Inferior 1 (process 144399) exited normally]
(gdb) 
```

Рис. 5: Проверка программы отладчиком

Для более подробного анализа программы добавляю брейкпоинт на метку `_start` и снова запускаю отладку (рис. 6).

```
irina_panyavkina@fedora:~/work/arch-pc/lab09

irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/study/2024... x

irina_panyavkina@vbox:~/work/arch-pc/lab09$ nasm -f elf -g -l lab9-2.lst lab9-2.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-2 lab9-2.o
irina_panyavkina@vbox:~/work/arch-pc/lab09$ gdb lab9-2
GNU gdb (Fedora Linux) 14.2-1.fc40
Copyright (C) 2023 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab9-2...
(gdb) run
Starting program: /home/irina_panyavkina/work/arch-pc/lab09/lab9-2

This GDB supports auto-downloading debuginfo from the following URLs:
<https://debuginfod.fedoraproject.org/>
Enable debuginfod for this session? (y or [n]) y
Debuginfod has been enabled.
To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
Hello, World!
[Inferior 1 (process 144399) exited normally]
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab9-2.asm, line 11.
(gdb) run
Starting program: /home/irina_panyavkina/work/arch-pc/lab09/lab9-2

Breakpoint 1, _start () at lab9-2.asm:11
11      mov eax, 4
(gdb) 
```

Рис. 6: Запуск отладчика с брейкпоинтом


```
irina_panyavkina@fedora:~/work/arch-pc/lab09
irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/arch-pc/lab... x
--Register group: general
eax 0x0 0 ecx 0x0 0
edx 0x0 0 ebx 0x0 0
esp 0xffffd060 0xffffd060 ebp 0x0 0
esi 0x0 0 edi 0x0 0
eip 0x8049000 0x8049000 <_start> eflags 0x202 [ IF ]
cs 0x23 35 ss 0x2b 43
ds 0x2b 43 es 0x2b 43
fs 0x0 0 gs 0x0 0

0x8049000 <_start> mov eax,0x4
0x8049005 <_start+5> mov ebx,ss
0x804900a <_start+10> mov ecx,0x804a000
0x804900f <_start+15> mov edx,0x8
0x8049014 <_start+20> int 0x80
0x8049016 <_start+22> mov eax,edx
0x804901b <_start+27> mov ebx,ecx
0x8049020 <_start+32> mov ecx,0x804a008
0x8049025 <_start+37> mov edx,ecx
0x804902a <_start+42> int 0x80
0x804902c <_start+44> mov ebx,ecx

native process 144432 In: _start
(gdb) tui reg general
(gdb) 
```

Рис. 8: Режим псевдографики

Добавление точек останова

Проверяю в режиме псевдографики, что брейкпоинт сохранился (рис. 9).

```
irina_panyavkina@fedora:~/work/arch-pc/lab09
irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/arch-pc/lab... x
--Register group: general
eax 0x0 0 ecx 0x0 0
edx 0x0 0 ebx 0x0 0
esp 0xffffd060 0xffffd060 ebp 0x0 0
esi 0x0 0 edi 0x0 0
eip 0x8049000 0x8049000 <_start> eflags 0x202 [ IF ]
cs 0x23 35 ss 0x2b 43
ds 0x2b 43 es 0x2b 43
fs 0x0 0 gs 0x0 0

0x8049000 <_start> mov eax,0x4
0x8049005 <_start+5> mov ebx,ss
0x804900a <_start+10> mov ecx,0x804a000
0x804900f <_start+15> mov edx,0x8
0x8049016 <_start+22> mov eax,edx
0x804901b <_start+27> mov ebx,ecx
0x8049020 <_start+32> mov ecx,0x804a008
0x8049025 <_start+37> mov edx,ecx
0x804902a <_start+42> int 0x80
0x804902c <_start+44> mov ebx,ecx

native process 144432 In: _start
(gdb) tui reg general
(gdb) info breakpoints
Num Type Disp Enb Address What
1 breakpoint keep y 0x8049000 lab09-2.asm:11
breakpoint already hit 1 time
(gdb) 
```

Рис. 9: Список брейкпоинтов

Устанавливаю ещё одну точку останова по адресу инструкции (рис. 10).


```
irina_panyavkina@fedora:~/work/arch-pc/lab09
irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/arch-pc/lab... x
Register group: general
eax 0x8 8 ecx 0x804a000 134520832
edx 0x8 8 ebx 0x1 1
esp 0xffffd060 0xffffd060 ebp 0x0 0
esi 0x0 0 edi 0x0 0
eip 0x8049016 0x8049016 <_start+22> eflags 0x202 [ IF ]
cs 0x23 35 ss 0x2b 43
ds 0x2b 43 es 0x2b 43
fs 0x0 0 gs 0x0 0

0x8049009 <_start+10> mov ecx,0x804a000
0x804900f <_start+15> mov edx,0x8
0x8049014 <_start+20> int 0x0
0x8049016 <_start+22> mov eax,0x4
0x804901d <_start+27> mov ebx,0x1
0x8049020 <_start+32> mov ecx,0x804a000
0x8049025 <_start+37> mov edx,0x7
0x804902a <_start+42> int 0x0
0x804902f <_start+44> mov eax,0x1
0x8049031 <_start+46> mov ebx,0x0

native process 144432 In: _start
esi 0x0 0
edi 0x0 0
eip 0x8049016 0x8049016 <_start+22>
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
(gdb) x/15b $msg1
0x8049000 <msg1>: "Hello, "
(gdb) x/15b 0x804a008
0x804a008 <msg2>: "World!\n034"
(gdb) 
```

Рис. 12: Просмотр содержимого переменных двумя способами

Меняю содержимое переменных по имени и адресу (рис. 13).

```
irina_panyavkina@fedora:~/work/arch-pc/lab09
irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/study/2024... x irina_panyavkina@fedora:~/work/arch-pc/lab... x
Register group: general
eax 0x8 8 ecx 0x804a000 134520832
edx 0x8 8 ebx 0x1 1
esp 0xffffd060 0xffffd060 ebp 0x0 0
esi 0x0 0 edi 0x0 0
eip 0x8049016 0x8049016 <_start+22> eflags 0x202 [ IF ]
cs 0x23 35 ss 0x2b 43
ds 0x2b 43 es 0x2b 43
fs 0x0 0 gs 0x0 0

0x8049009 <_start+10> mov ecx,0x804a000
0x804900f <_start+15> mov edx,0x8
0x8049014 <_start+20> int 0x0
0x8049016 <_start+22> mov eax,0x4
0x804901d <_start+27> mov ebx,0x1
0x8049020 <_start+32> mov ecx,0x804a000
0x8049025 <_start+37> mov edx,0x7
0x804902a <_start+42> int 0x0
0x804902f <_start+44> mov eax,0x1
0x8049031 <_start+46> mov ebx,0x0

native process 144432 In: _start
(gdb) <msg1>: "Hello, "
(gdb) x/15b 0x804a008
0x804a008 <msg2>: "World!\n034"
(gdb) set (char)msg1="h"
(gdb) x/15b $msg1
0x8049000 <msg1>: "hello, "
(gdb) set (char)msg2="x"
(gdb) x/15b $msg2
0x804a008 <msg2>: "xorld!\n034"
(gdb) 
```

Рис. 13: Изменение содержимого переменных двумя способами

Вывожу в различных форматах значение регистра edx (рис. 14).

The screenshot shows the GDB interface with the 'Register group: general' window. The registers are listed with their values: eax (0x0), ebx (0x0), ecx (0x804a000), edx (0x0), esp (0xffffd060), esi (0x0), edi (0x0), eip (0x8049016), eflags (0x202), cs (0x23), ds (0x2b), fs (0x0), ss (0x2b), es (0x2b), gs (0x0). The assembly window shows instructions: 0x804900c <start+10> mov ecx, 0x804a000; 0x804900f <start+15> mov ebx, 0x0; 0x8049016 <start+22> int \$0x0; 0x8049016 <start+22> mov eax, 0x0; 0x804901c <start+27> mov ebx, 0x0; 0x8049020 <start+32> mov ecx, 0x804a000; 0x8049025 <start+37> mov edi, 0x7; 0x804902a <start+42> int \$0x0; 0x804902c <start+44> mov eax, 0x1. The console shows: native process 144432 In: _start; \$1 = 1000; (gdb) p/t \$ecx; \$2 = 100000001001010000000000000000; (gdb) p/s \$edx; \$3 = 8; (gdb) p/t \$edx; \$4 = 1000; (gdb) p/x \$edx; \$5 = 0x8; (gdb) []

Рис. 14: Просмотр значения регистра разными представлениями

С помощью команды set меняю содержимое регистра ebx (рис. 15).

The screenshot shows the GDB interface with the 'Register group: general' window. The registers are listed with their values: eax (0x0), ebx (0x2), ecx (0x804a000), edx (0x0), esp (0xffffd060), esi (0x0), edi (0x0), eip (0x8049016), eflags (0x202), cs (0x23), ds (0x2b), fs (0x0), ss (0x2b), es (0x2b), gs (0x0). The assembly window shows instructions: 0x804900c <start+10> mov ecx, 0x804a000; 0x804900f <start+15> mov ebx, 0x0; 0x8049016 <start+22> int \$0x0; 0x8049016 <start+22> mov eax, 0x0; 0x804901c <start+27> mov ebx, 0x1; 0x8049020 <start+32> mov ecx, 0x804a000; 0x8049025 <start+37> mov edi, 0x7; 0x804902a <start+42> int \$0x0; 0x804902c <start+44> mov eax, 0x1. The console shows: native process 144432 In: _start; \$2 = 0x2; (gdb) set \$ebx=2; (gdb) p/s; \$6 = 8; (gdb) p/s \$ebx; \$7 = 50; (gdb) set \$ebx=2; (gdb) p/s \$ebx; \$8 = 2; (gdb) []

Рис. 15: Подготовка новой программы

Обработка аргументов командной строки в GDB

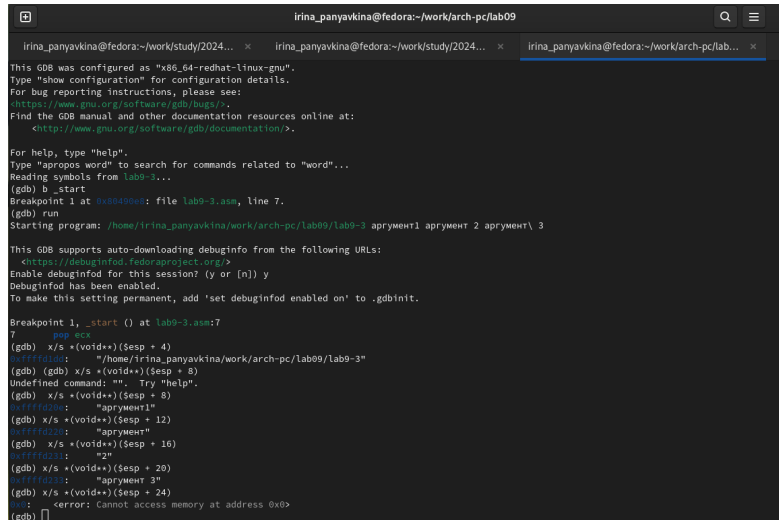
Копирую программу из предыдущей лабораторной работы в текущий каталог и создаю исполняемый файл с файлом листинга и отладки (рис. 16).

The screenshot shows the terminal with the following commands: irina_panyavkina@vbox: /work/arch-pc/lab09\$ cp ~/work/arch-pc/lab08/lab8-2.asm ~/work/arch-pc/lab09/lab9-3.asm; irina_panyavkina@vbox: /work/arch-pc/lab09\$ nasm -f elf -g -l lab9-3.lst lab9-3.asm; irina_panyavkina@vbox: /work/arch-pc/lab09\$ ld -m elf_i386 -o lab9-3 lab9-3.o; irina_panyavkina@vbox: /work/arch-pc/lab09\$ []

Рис. 16: Подготовка новой программы

Запускаю программу с режиме отладки с указанием аргументов, указываю брейкпоинт и запускаю отладку. Проверяю работу стека, изменяя аргумент команды просмотра регистра esp на +4, число обусловлено разрядностью

системы, а указатель void занимает как раз 4 байта, ошибка при аргументе +24 означает, что аргументы на вход программы закончились. (рис. 17).



```
irina_panyavkina@fedora:~/work/arch-pc/lab09
This GDB was configured as "x86_64-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
  https://www.gnu.org/software/gdb/bugs/.
Find the GDB manual and other documentation resources online at:
  <http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab9-3...
(gdb) b _start
Breakpoint 1 at 0x2040000: file lab9-3.asm, line 7.
(gdb) run
Starting program: /home/irina_panyavkina/work/arch-pc/lab09/lab9-3 аргумент1 аргумент2 аргумент\3
This GDB supports auto-downloading debuginfo from the following URLs:
  <https://debuginfod.fedoraproject.org>
Enable debuginfod for this session? (y or [n]) y
Debuginfod has been enabled.
To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
Breakpoint 1, _start () at lab9-3.asm:7
7      pop ecx
(gdb) x/s *(void**)($esp + 4)
0x00000000: "/home/irina_panyavkina/work/arch-pc/lab09/lab9-3"
(gdb) (gdb) x/s *(void**)($esp + 8)
0x00000000: ""
Undefined command: "".  Try "help".
(gdb) x/s *(void**)($esp + 8)
0x00000000: "аргумент1"
(gdb) x/s *(void**)($esp + 12)
0x00000000: "аргумент"
(gdb) x/s *(void**)($esp + 16)
0x00000000: "2"
(gdb) x/s *(void**)($esp + 20)
0x00000000: "аргумент 3"
(gdb) x/s *(void**)($esp + 24)
0x00000000: <error: Cannot access memory at address 0x0>
(gdb)
```

Рис. 17: Проверка работы стека

Выполнение заданий для самостоятельной работы

1. Меняю программу самостоятельной части предыдущей лабораторной работы с использованием подпрограммы (рис. 18).

```

*~/work/arch-pc/lab09/lab9-4.asm - Mousepad
File Edit Search View Document Help
1 %include 'in_out.asm'
2
3 SECTION .data
4 msg_func db "Функция: f(x) = 6x + 13", 0
5 msg_result db "Результат: ", 0
6
7 SECTION .text
8 GLOBAL _start
9
10 _start:
11 mov eax, msg_func
12 call sprintf
13
14 pop ecx
15 pop edx
16 sub ecx, 1
17 mov esi, 0
18
19 next:
20 cmp ecx, 0
21 jz _end
22 pop eax
23 call atoi
24
25 call _calculate_fx
26
27 add esi, eax
28
29 loop next
30
31 _end:
32 mov eax, msg_result
33 call sprintf
34 mov eax, esi
35 call iprintLF
36 call quit
37
38 _calculate_fx
39 mov ebx, 6
40 mul ebx
41 add eax, 13
42 ret

```

Рис. 18: Изменённая программа предыдущей лабораторной работы

Код программы:

```
%include 'in_out.asm'

SECTION .data
msg_func db "Функция: f(x) = 6x + 13", 0
msg_result db "Результат: ", 0

SECTION .text
GLOBAL _start

_start:
mov eax, msg_func
call sprintf
```

```

pop ecx
pop edx
sub ecx, 1
mov esi, 0

next:
cmp ecx, 0
jz _end
pop eax
call atoi

call _calculate_fx

add esi, eax

loop next

_end:
mov eax, msg_result
call sprint
mov eax, esi
call iprintLF
call quit

_calculate_fx
mov ebx, 6
mul ebx
add eax, 13
ret

```

2. Запускаю программу в режиме отладчика и пошагово через «stepi» просматриваю изменение значений регистров через «info registers». При выполнении инструкции «mul ecx» можно заметить, что результат умножения записывается в регистр eax, но также меняет и edx. Значение регистра ebx не обновляется напрямую, поэтому в результате программа неверно высчитывает функцию (рис. 19).

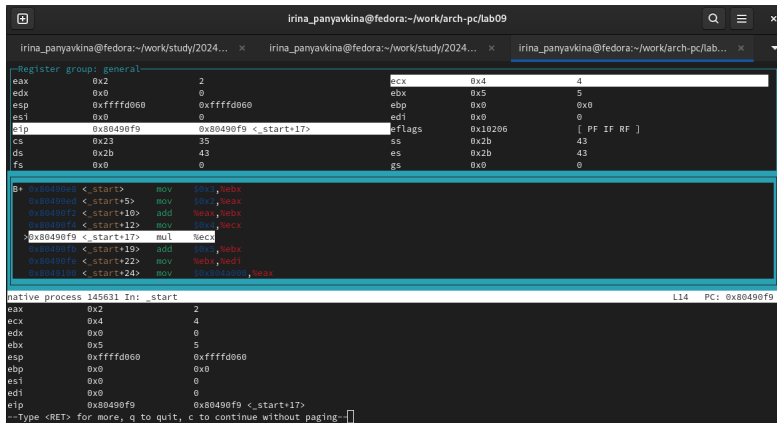


Рис. 19: Поиск ошибки в программе через пошаговую отладку

Исправляю найденную ошибку, теперь программа правильно считает значение функции (рис. 20).

```

irina_panyavkina@vbox:~/work/arch-pc/lab09$ nasm -f elf lab9-5.asm
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-5 lab9-5.o
irina_panyavkina@vbox:~/work/arch-pc/lab09$ ./lab9-5
Результат: 25
irina_panyavkina@vbox:~/work/arch-pc/lab09$

```

Рис. 20: Проверка корректировок в программе

Код изменённой программы:

```

#include 'in_out.asm'

SECTION .data
div: DB 'Результат: ', 0

SECTION .text
GLOBAL _start
_start:

mov ebx, 3
mov eax, 2
add ebx, eax
mov eax, ebx
mov ecx, 4
mul ecx
add eax, 5
mov edi, eax

mov eax, div

```

```
call sprint  
mov eax, edi  
call iprintLF
```

```
call quit
```

5 Выводы

В ходе выполнения лабораторной работы, я приобрела навыки написания программ с использованием подпрограмм и познакомилась с методами отладки при помощи GDB и его основными возможностями.

Список литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.Org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658>.
6. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
10. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.
11. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
12. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.
13. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.

14. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix.
15. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
16. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science).