

Отчёт по лабораторной работе

Дисциплина: Архитектура ЭВМ

Перегудов Александр Вадимович

Содержание

1	Цель работы	6
2	Задание	7
3	Теоретическое введение	8
4	Выполнение лабораторной работы	9
5	Выводы	28
	Список литературы	29

Список иллюстраций

4.1	Создание каталога и файла	9
4.2	Текст программы	9
4.3	Результат работы программы	10
4.4	Изменённый текст программы	10
4.5	Результат работы программы	11
4.6	Создание файла	11
4.7	Текст программы	11
4.8	Трансляция и компоновка	12
4.9	Отладчик gdb	12
4.10	Запуск программы в отладчике	12
4.11	Установка брейкпоинта	12
4.12	Дисассимилированный код	13
4.13	Команда set disassembly-flavor intel	14
4.14	Команда	14
4.15	Режим псевдографики	15
4.16	Режим псевдографики с отображением регистров	15
4.17	Установленные точки останова	16
4.18	Установленная точка останова	16
4.19	Команды stepi	17
4.20	Команда info registers	17
4.21	Значения регистров	18
4.22	Значение переменной msg1	18
4.23	Значение переменной msg2	18
4.24	Инструкция mov ecx, msg2	18
4.25	Команда set	19
4.26	Команда set	19
4.27	Значение регистра edx	20
4.28	Команда set	20
4.29	Команда continue и quit	21
4.30	Копирование файла	21
4.31	Трансляция и компоновка	21
4.32	Исполняемый файл в отладчике	21
4.33	Точка останова и запуск	22
4.34	Значение esp	22
4.35	Значения позиций стека	22
4.36	Копирование	22
4.37	Текст программы	23

4.38	Результат работы программы	24
4.39	Создание файла	24
4.40	Текст программы	24
4.41	Трансляция и компоновка	25
4.42	Программа в оболочке GDB	25
4.43	Точка останова и переключение отображения	25
4.44	Команда layout	25
4.45	Режим псевдографики с отображением регистров и запуск	26
4.46	Команды stepi	26
4.47	Изменённый текст программы	27
4.48	Результат работы программы	27

Список таблиц

3.1	Описание некоторых каталогов файловой системы GNU Linux . .	8
-----	---	---

1 Цель работы

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

2 Задание

Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы. Например, в табл. 3.1 приведено краткое описание стандартных каталогов Unix.

Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux

Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно об Unix см. в [1–6].

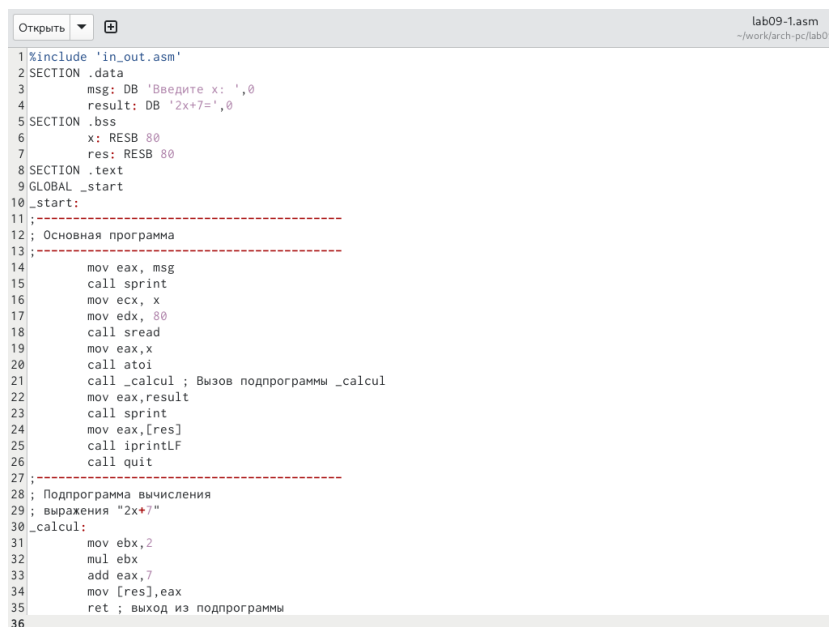
4 Выполнение лабораторной работы

1. Создал каталог для выполнения лабораторной работы № 9, перешёл в него и создал файл lab09-1.asm (рис. 4.1).

```
avperegudov@dk4n63 ~ $ mkdir ~/work/arch-pc/lab09
avperegudov@dk4n63 ~ $ cd ~/work/arch-pc/lab09
avperegudov@dk4n63 ~/work/arch-pc/lab09 $ touch lab09-1.asm
avperegudov@dk4n63 ~/work/arch-pc/lab09 $
```

Рис. 4.1: Создание каталога и файла

2. Ввёл в файл lab09-1.asm текст программы из листинга 9.1 (рис. 4.2).



```
lab09-1.asm
~/work/arch-pc/lab09

1 %include 'in_out.asm'
2 SECTION .data
3     msg: DB 'Введите x: ',0
4     result: DB '2x+7=',0
5 SECTION .bss
6     x: RESB 80
7     res: RESB 80
8 SECTION .text
9 GLOBAL _start
10 _start:
11 ;-----
12 ; Основная программа
13 ;-----
14     mov eax, msg
15     call sprint
16     mov ecx, x
17     mov edx, 80
18     call sread
19     mov eax, x
20     call atoi
21     call _calcul ; Вызов подпрограммы _calcul
22     mov eax, result
23     call sprint
24     mov eax, [res]
25     call iprintfLF
26     call quit
27 ;-----
28 ; Подпрограмма вычисления
29 ; выражения "2x+7"
30 _calcul:
31     mov ebx, 2
32     mul ebx
33     add eax, 7
34     mov [res], eax
35     ret ; выход из подпрограммы
36
```

Рис. 4.2: Текст программы

3. Создал исполняемый файл и проверил его работу (рис. 4.3).

```
avperegudov@dk4n63 ~/work/arch-pc/lab09 $ nasm -f elf lab09-1.asm
avperegudov@dk4n63 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-1 lab09-1.o
avperegudov@dk4n63 ~/work/arch-pc/lab09 $ ./lab09-1
Введите x: 3
2x+7=13
```

Рис. 4.3: Результат работы программы

4. Изменил текст программы, добавив подпрограмму `_subcalcul` в подпрограмму `_calcul`, для вычисления выражения $f(g(x))$, где x вводится с клавиатуры, $f(x) = 2x + 7$, $g(x) = 3x - 1$ (рис. 4.4).

```
Открыть  lab09-1.asm
~/work/arch-pc/lab09

1 %include 'in_out.asm'
2 SECTION .data
3     msg: DB 'Введите x: ',0
4     result: DB 'y=3x-1 2y+7=',0
5 SECTION .bss
6     x: RESB 80
7     res: RESB 80
8 SECTION .text
9 GLOBAL _start
0 _start:
1 ; -----
2 ; Основная программа
3 ; -----
4     mov eax, msg
5     call sprint
6     mov ecx, x
7     mov edx, 80
8     call sread
9     mov eax, x
10    call atoi
11    call _calcul ; Вызов подпрограммы _calcul
12    mov eax, result
13    call sprint
14    mov eax, [res]
15    call iprintLF
16    call quit
17 ; -----
18 ; Подпрограмма вычисления
19 ; выражения "2x+7"
20 _calcul:
21    call _subcalcul
22    mov ebx, 2
23    mul ebx
24    add eax, 7
25    mov [res], eax
26    ret ; выход из подпрограммы
27 _subcalcul:
28    mov ebx, 3
29    mul ebx
30    sub eax, 1
31    ret
```

Рис. 4.4: Изменённый текст программы

5. Создал исполняемый файл и проверил его работу (рис. 4.5).

```
avperegudov@dk4n63 ~/work/arch-pc/lab09 $ nasm -f elf lab09-1.asm
avperegudov@dk4n63 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-1 lab09-1.o
avperegudov@dk4n63 ~/work/arch-pc/lab09 $ ./lab09-1
Введите x: 3
y=3x-1 2y+7=23
```

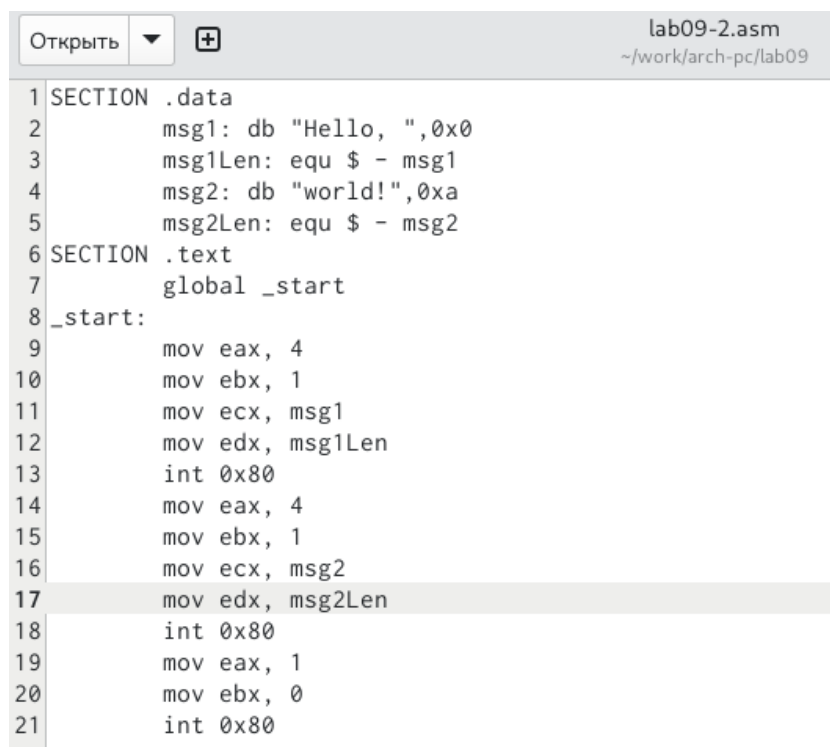
Рис. 4.5: Результат работы программы

6. Создал файл lab09-2.asm (рис. 4.6).

```
avperegudov@dk4n63 ~/work/arch-pc/lab09 $ touch lab09-2.asm
```

Рис. 4.6: Создание файла

7. Ввёл в файл lab09-2.asm текст программы из листинга 9.2 (рис. 4.7).



```
lab09-2.asm
~/work/arch-pc/lab09

1 SECTION .data
2     msg1: db "Hello, ",0x0
3     msg1Len: equ $ - msg1
4     msg2: db "world!",0xa
5     msg2Len: equ $ - msg2
6 SECTION .text
7     global _start
8 _start:
9     mov eax, 4
10    mov ebx, 1
11    mov ecx, msg1
12    mov edx, msg1Len
13    int 0x80
14    mov eax, 4
15    mov ebx, 1
16    mov ecx, msg2
17    mov edx, msg2Len
18    int 0x80
19    mov eax, 1
20    mov ebx, 0
21    int 0x80
```

Рис. 4.7: Текст программы

8. Создал исполняемый файл (рис. 4.8).

```
nasm -f elf -g -l lab09-2.lst lab09-2.asm
ld -m elf_i386 -o lab09-2 lab09-2.o
```

Рис. 4.8: Трансляция и компоновка

9. Загрузил исполняемый файл в отладчик gdb (рис. 4.9).

```
avperegudov@dk4n63 ~/work/arch-pc/lab09 $ gdb lab09-2
GNU gdb (Gentoo 12.1 vanilla) 12.1
Copyright (C) 2022 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-pc-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://bugs.gentoo.org/>.
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 lab09-2...
(gdb) █
```

Рис. 4.9: Отладчик gdb

10. Проверил работу программы, запустив ее в оболочке GDB с помощью команды run (рис. 4.10).

```
(gdb) run
Starting program: /afs/.dk.sci.pfu.edu.ru/home/a/v/avperegudov/work/arch-pc/lab09/lab09-2
Hello, world!
[Inferior 1 (process 33719) exited normally]
(gdb) █
```

Рис. 4.10: Запуск программы в отладчике

11. Установил точку останова на метку _start и запустил её. (рис. 4.11).

```
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab09-2.asm, line 9.
(gdb) run
Starting program: /afs/.dk.sci.pfu.edu.ru/home/a/v/avperegudov/work/arch-pc/lab09/lab09-2

Breakpoint 1, _start () at lab09-2.asm:9
9      mov eax, 4
(gdb) █
```

Рис. 4.11: Установка брейкпоинта

12. Посмотрел дисассимилированный код программы начиная с метки `_start` (рис. 4.12).

```
(gdb) disassemble _start
Dump of assembler code for function _start:
=> 0x08049000 <+0>:      mov     $0x4,%eax
    0x08049005 <+5>:      mov     $0x1,%ebx
    0x0804900a <+10>:     mov     $0x804a000,%ecx
    0x0804900f <+15>:     mov     $0x8,%edx
    0x08049014 <+20>:     int     $0x80
    0x08049016 <+22>:     mov     $0x4,%eax
    0x0804901b <+27>:     mov     $0x1,%ebx
    0x08049020 <+32>:     mov     $0x804a008,%ecx
    0x08049025 <+37>:     mov     $0x7,%edx
    0x0804902a <+42>:     int     $0x80
    0x0804902c <+44>:     mov     $0x1,%eax
    0x08049031 <+49>:     mov     $0x0,%ebx
    0x08049036 <+54>:     int     $0x80
End of assembler dump.
```

Рис. 4.12: Дисассимилированный код

13. Переключился на отображение команд с Intel'овским синтаксисом (рис. 4.13).

```

(gdb) set disassembly-flavor intel
(gdb) disassemble _start
Dump of assembler code for function _start:
=> 0x08049000 <+0>:      mov     eax,0x4
    0x08049005 <+5>:      mov     ebx,0x1
    0x0804900a <+10>:     mov     ecx,0x804a000
    0x0804900f <+15>:     mov     edx,0x8
    0x08049014 <+20>:     int     0x80
    0x08049016 <+22>:     mov     eax,0x4
    0x0804901b <+27>:     mov     ebx,0x1
    0x08049020 <+32>:     mov     ecx,0x804a008
    0x08049025 <+37>:     mov     edx,0x7
    0x0804902a <+42>:     int     0x80
    0x0804902c <+44>:     mov     eax,0x1
    0x08049031 <+49>:     mov     ebx,0x0
    0x08049036 <+54>:     int     0x80
End of assembler dump.

```

Рис. 4.13: Команда set disassembly-flavor intel

14. Включил режим псевдографики (рис. 4.15).

```

(gdb) layout asm

```

Рис. 4.14: Команда

```
B+> 0x8049000 <_start>      mov     eax,0x4
      0x8049005 <_start+5>    mov     ebx,0x1
      0x804900a <_start+10>   mov     ecx,0x804a000
      0x804900f <_start+15>   mov     edx,0x8
      0x8049014 <_start+20>   int     0x80
      0x8049016 <_start+22>   mov     eax,0x4
      0x804901b <_start+27>   mov     ebx,0x1
      0x8049020 <_start+32>   mov     ecx,0x804a008
      0x8049025 <_start+37>   mov     edx,0x7
      0x804902a <_start+42>   int     0x80
      0x804902c <_start+44>   mov     eax,0x1
      0x8049031 <_start+49>   mov     ebx,0x0
      0x8049036 <_start+54>   int     0x80

native process 33733 In: _start
(gdb) 
```

Рис. 4.15: Режим псевдографики

15. Включил режим псевдографики с отображением регистров (рис. 4.16).

```
[ Register Values Unavailable ]

B+> 0x8049000 <_start>      mov     eax,0x4
      0x8049005 <_start+5>    mov     ebx,0x1
      0x804900a <_start+10>   mov     ecx,0x804a000
      0x804900f <_start+15>   mov     edx,0x8
      0x8049014 <_start+20>   int     0x80
      0x8049016 <_start+22>   mov     eax,0x4
      0x804901b <_start+27>   mov     ebx,0x1

native process 33733 In: _start
(gdb) layout regs
(gdb) 
```

Рис. 4.16: Режим псевдографики с отображением регистров

16. Проверил установленные точки останова с помощью команды `info breakpoints` (рис. 4.17).

```
(gdb) info breakpoints
Num      Type           Disp Enb Address      What
1        breakpoint     keep y   0x08049000  lab09-2.asm:9
```

Рис. 4.17: Установленные точки останова

17. Установил еще одну точку останова по адресу инструкции и посмотрел информацию о всех установленных точках останова (рис. 4.18).

```
b+ 0x8049000 <_start>      mov     eax,0x4
    0x8049005 <_start+5>    mov     ebx,0x1
    0x804900a <_start+10>   mov     ecx,0x804a000
    0x804900f <_start+15>   mov     edx,0x8
    0x8049014 <_start+20>   int     0x80
    0x8049016 <_start+22>   mov     eax,0x4
    0x804901b <_start+27>   mov     ebx,0x1
    0x8049020 <_start+32>   mov     ecx,0x804a008
    0x8049025 <_start+37>   mov     edx,0x7
    0x804902a <_start+42>   int     0x80
    0x804902c <_start+44>   mov     eax,0x1
b+ 0x8049031 <_start+49>   mov     ebx,0x0
    0x8049036 <_start+54>   int     0x80

exec No process in:
(gdb) break *0x8049031
Breakpoint 2 at 0x8049031: file lab09-2.asm, line 20.
(gdb) i b
Num      Type           Disp Enb Address      What
1        breakpoint     keep y   0x08049000  lab09-2.asm:9
2        breakpoint     keep y   0x08049031  lab09-2.asm:20
(gdb)
```

Рис. 4.18: Установленная точка останова

18. Выполнял 5 инструкций с помощью команды `stepi` (рис. 4.19).


```
B+ 0x8049000 <_start>      mov     eax,0x4
    0x8049005 <_start+5>    mov     ebx,0x1
    0x804900a <_start+10>   mov     ecx,0x804a000
    0x804900f <_start+15>   mov     edx,0x8
    0x8049014 <_start+20>   int     0x80
> 0x8049016 <_start+22>   mov     eax,0x4
    0x804901b <_start+27>   mov     ebx,0x1
    0x8049020 <_start+32>   mov     ecx,0x804a008
    0x8049025 <_start+37>   mov     edx,0x7
    0x804902a <_start+42>   int     0x80
    0x804902c <_start+44>   mov     eax,0x1
b+ 0x8049031 <_start+49>   mov     ebx,0x0
    0x8049036 <_start+54>   int     0x80

native process 33936 In: _start

Breakpoint 1, _start () at lab09-2.asm:9
(gdb) stepi
(gdb) stepi
(gdb) stepi
(gdb) stepi
(gdb) stepi
(gdb) stepi
(gdb) █
```

Рис. 4.19: Команды stepi

19. Посмотрел содержимое регистров (рис. 4.21).

```
(gdb) info registers █
```

Рис. 4.20: Команда info registers

```

native process 33936 In: _start
eax          0x8          8
ecx          0x804a000    134520832
edx          0x8          8
ebx          0x1          1
esp          0xffffc300   0xffffc300
ebp          0x0          0x0
esi          0x0          0
--Type <RET> for more, q to quit, c to continue without paging--

```

Рис. 4.21: Значения регистров

20. Посмотрел значение переменной msg1 по имени (рис. 4.22).

```

(gdb) x/1sb &msg1
0x804a000 <msg1>:      "Hello, "
(gdb) 

```

Рис. 4.22: Значение переменной msg1

21. Посмотрел значение переменной msg2 по адресу (рис. 4.23).

```

(gdb) x 0x804a008
0x804a008 <msg2>:      "world!\n\034"
(gdb) 

```

Рис. 4.23: Значение переменной msg2

22. Посмотрел инструкцию mov ecx,msg2 которая записывает в регистр ecx адрес переменной msg2 (рис. 4.24).

```

(gdb) x 0x8049020
0x8049020 <_start+32>:  "\271\b\240\004\b\272\a"

```

Рис. 4.24: Инструкция mov ecx, msg2

23. Изменил первый символ переменной msg1 (рис. 4.25).

```
(gdb) set {char}&msg1='h'
(gdb) x/1sb &msg1
0x804a000 <msg1>:      "hello, "
```

Рис. 4.25: Команда set

24. Изменил первый символ переменной msg2 (рис. 4.26).

```
(gdb) set {char}&msg2='x'
(gdb) x/1sb &msg2
0x804a008 <msg2>:      "xorld!\n\034"
```

Рис. 4.26: Команда set

25. Вывел в различных форматах значение регистра edx (рис. 4.27).

```
(gdb) p/s $edx
$1 = 8

(gdb) p/t $edx
$2 = 1000

(gdb) p/x $edx
$3 = 0x8
```

Рис. 4.27: Значение регистра edx

26. Изменил значение регистра ebx (рис. 4.28).

```
(gdb) set $ebx='2'
(gdb) p/s $ebx
$4 = 50
```

Рис. 4.28: Команда set

27. Завершил выполнение программы с помощью команды continue и вышел из GDB с помощью команды quit (рис. 4.29).

```
(gdb) continue
Continuing.
world!
(gdb) eor 1 (process 34875) exited normally]
Ambiguous command "e": echo, edit, en, enable, end, eval, exec-file, exit, explore, expression.
(gdb) quit
```

Рис. 4.29: Команда continue и quit

28. Скопировал файл lab8-2.asm, созданный при выполнении лабораторной работы №8 (рис. 4.30).

```
avperegudov@Study:~/work/arch-pc/lab09/report$ cp ~/work/arch-pc/lab08/lab8-2.asm ~/work/arch-pc/lab09/lab09-3.asm
avperegudov@Study:~/work/arch-pc/lab09/report$
```

Рис. 4.30: Копирование файла

29. Создал исполняемый файл (рис. 4.31).

```
avperegudov@Study:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-3.lst lab09-3.asm
avperegudov@Study:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-3 lab09-3.o
```

Рис. 4.31: Трансляция и компоновка

30. Загрузил исполняемый файл в отладчик, указав аргументы (рис. 4.32).

```
avperegudov@Study:~/work/arch-pc/lab09$ gdb --args lab09-3 аргумент1 аргумент 2 'аргумент 3'
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 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-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 lab09-3...
(gdb)
```

Рис. 4.32: Исполняемый файл в отладчике

31. Установил точку останова перед первой инструкцией в программе и запустил ее (рис. 4.33).

```
(gdb) b _start
Breakpoint 1 at 0x80490e8: file lab09-3.asm, line 5.
(gdb) run
Starting program: /home/avperegudov/work/arch-pc/lab09/lab09-3 аргумент1 аргумен
т 2 аргумент\ 3
Breakpoint 1, _start () at lab09-3.asm:5
5      pop ecx ; Извлекаем из стека в 'ecx' количество
(gdb) |
```

Рис. 4.33: Точка останова и запуск

32. Посмотрел значение регистра esp (рис. 4.34).

```
(gdb) x/x $esp
0xffffd080: 0x00000005
(gdb) |
```

Рис. 4.34: Значение esp

33. Посмотрел остальные позиции стека (рис. 4.35).

```
(gdb) x/s *(void**)(esp + 4)
0xffffd268: "/home/avperegudov/work/arch-pc/lab09/lab09-3"
(gdb) x/s *(void**)(esp + 8)
0xffffd295: "аргумент1"
(gdb) x/s *(void**)(esp + 12)
0xffffd2a7: "аргумент"
(gdb) x/s *(void**)(esp + 16)
0xffffd2b8: "2"
(gdb) x/s *(void**)(esp + 20)
0xffffd2ba: "аргумент 3"
(gdb) x/s *(void**)(esp + 24)
0x0: <error: Cannot access memory at address 0x0>
(gdb) |
```

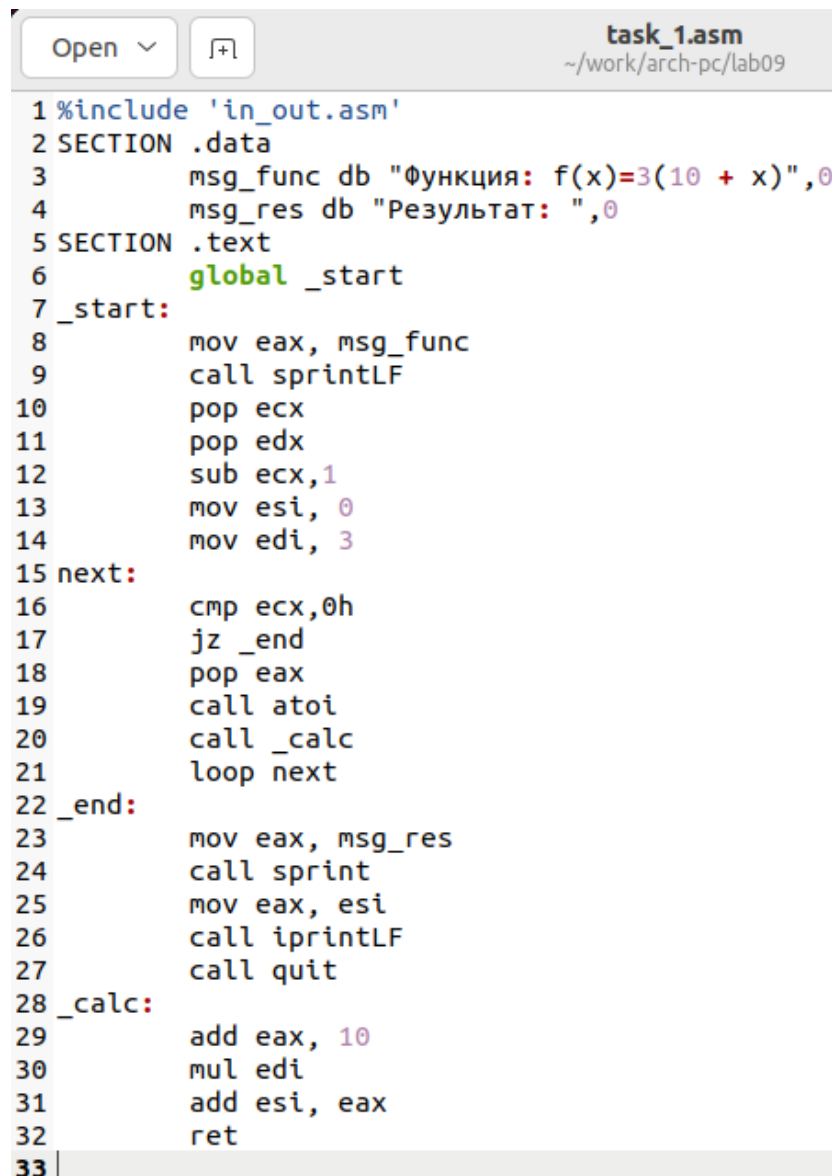
Рис. 4.35: Значения позиций стека

34. Скопировал программу из лабораторной работы № 8 (рис. 4.36).

```
avperegudov@Study:~/work/arch-pc/lab09$ cp ~/work/arch-pc/lab08/task.asm ~/work/
arch-pc/lab09/task_1.asm
```

Рис. 4.36: Копирование

35. Переписал программу реализовав вычисление значения функции $f(x)$ как подпрограмму (рис. 4.37).



```
task_1.asm
~/work/arch-pc/lab09

1 %include 'in_out.asm'
2 SECTION .data
3     msg_func db "Функция: f(x)=3(10 + x)",0
4     msg_res db "Результат: ",0
5 SECTION .text
6     global _start
7 _start:
8     mov eax, msg_func
9     call sprintf
10    pop ecx
11    pop edx
12    sub ecx,1
13    mov esi, 0
14    mov edi, 3
15 next:
16    cmp ecx,0h
17    jz _end
18    pop eax
19    call atoi
20    call _calc
21    loop next
22 _end:
23    mov eax, msg_res
24    call sprintf
25    mov eax, esi
26    call iprintLF
27    call quit
28 _calc:
29    add eax, 10
30    mul edi
31    add esi, eax
32    ret
33
```

Рис. 4.37: Текст программы

36. Создал исполняемый файл и проверил его работу с параметрами (рис. 4.38).

```
avperegudov@Study:~/work/arch-pc/lab09$ nasm -f elf task_1.asm
avperegudov@Study:~/work/arch-pc/lab09$ ld -m elf_i386 -o task_1 task_1.o
avperegudov@Study:~/work/arch-pc/lab09$ ./task_1 9 3 2
Функция: f(x)=3(10 + x)
Результат: 132
```

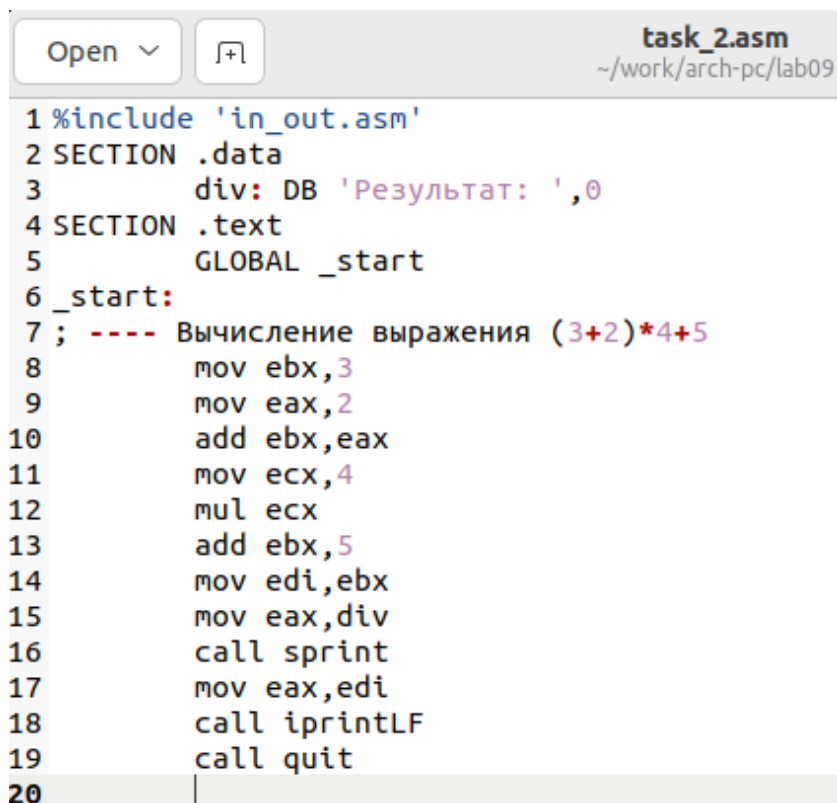
Рис. 4.38: Результат работы программы

37. Создал файл task_2.asm (рис. 4.39).

```
avperegudov@Study:~/work/arch-pc/lab09$ touch task_2.asm
```

Рис. 4.39: Создание файла

38. Ввёл в файл task_2.asm текст программы из листинга 9.3 (рис. 4.40).



```
task_2.asm
~/work/arch-pc/lab09

1 %include 'in_out.asm'
2 SECTION .data
3     div: DB 'Результат: ',0
4 SECTION .text
5     GLOBAL _start
6 _start:
7 ; ---- Вычисление выражения (3+2)*4+5
8     mov ebx,3
9     mov eax,2
10    add ebx,eax
11    mov ecx,4
12    mul ecx
13    add ebx,5
14    mov edi,ebx
15    mov eax,div
16    call sprint
17    mov eax,edi
18    call iprintfLF
19    call quit
20
```

Рис. 4.40: Текст программы

39. Создал исполняемый файл (рис. 4.41).


```
avperegudov@Study:~/work/arch-pc/lab09$ nasm -f elf -g -l task_2.lst task_2.asm
avperegudov@Study:~/work/arch-pc/lab09$ ld -m elf_i386 -o task_2 task_2.o
```

Рис. 4.41: Трансляция и компоновка

40. Запустил программу в оболочке GDB (рис. 4.42).

```
avperegudov@Study:~/work/arch-pc/lab09$ gdb task_2
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 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-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 task_2...
(gdb) █
```

Рис. 4.42: Программа в оболочке GDB

41. Установил точку останова на метку `_start` и переключился на отображение команд с Intel'овским синтаксисом (рис. 4.43).

```
(gdb) break _start
Breakpoint 1 at 0x80490e8: file task_2.asm, line 8.
(gdb) set disassembly-flavor intel
```

Рис. 4.43: Точка останова и переключение отображения

42. Включил режим псевдографики с отображением регистров (рис. 4.44).

```
(gdb) layout regs █
```

Рис. 4.44: Команда layout

43. Запустил программу в оболочке (рис. 4.45).

```

Register group: general
eax      0x0      0
ecx      0x0      0
edx      0x0      0
ebx      0x0      0
esp      0xffffd0d0 0xffffd0d0

6  _start:
7  : ---- Вычисление выражения (3+2)*4+5
B+> 8  mov ebx,3
9      mov eax,2
10     add ebx,eax
11     mov ecx,4
12     mul ecx

native process 5266 In: _start L8 PC: 0x80490e8
(gdb) run
Starting program: /home/avperegudov/work/arch-pc/lab09/task_2

```

Рис. 4.45: Режим псевдографики с отображением регистров и запуск

44. Ввёл 5 команд stepi и обнаружил что умножается не тот регистор (рис. 4.46).

```

Register group: general
eax      0x8      8
ecx      0x4      4
edx      0x0      0
ebx      0x5      5
esp      0xffffd0d0 0xffffd0d0

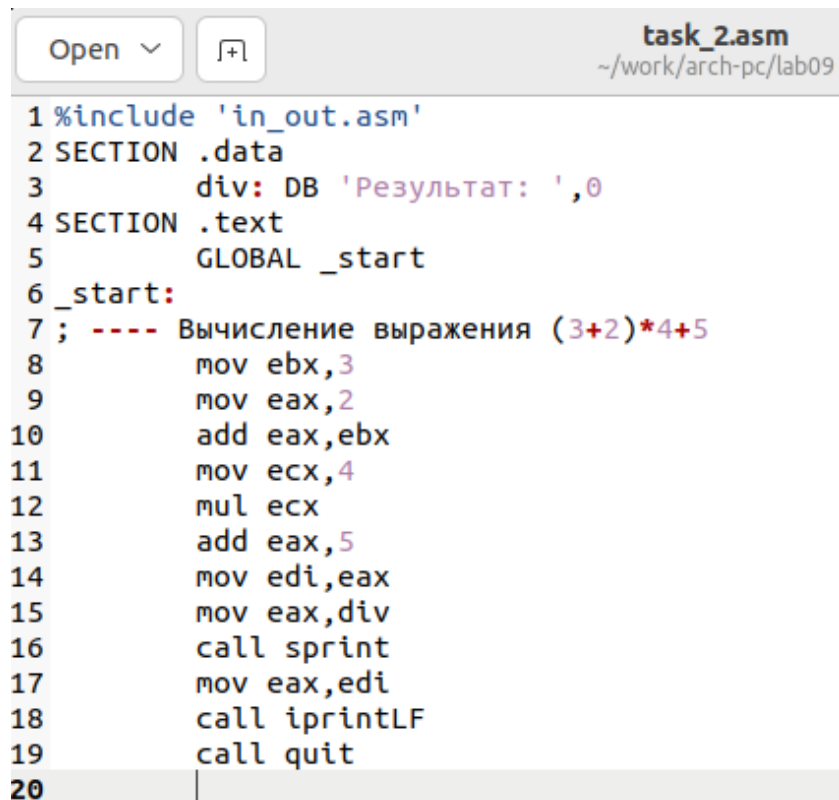
task_2.asm
9      mov eax,2
10     add ebx,eax
11     mov ecx,4
12     mul ecx
> 13     add ebx,5
14     mov edi,ebx
15     mov eax,div

native process 5303 In: _start L13 PC: 0x80490fb
Breakpoint 1, _start () at task_2.asm:8
(gdb) stepi
(gdb) stepi
(gdb) stepi
(gdb) stepi
(gdb) stepi
(gdb)

```

Рис. 4.46: Команды stepi

45. Изменил текст программы, приведя её в рабочее состояние (рис. 4.47).

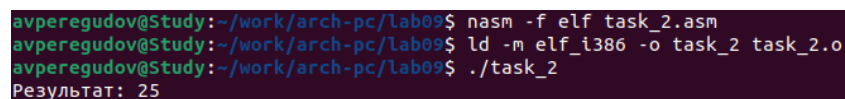


```
task_2.asm
~/work/arch-pc/lab09

1 %include 'in_out.asm'
2 SECTION .data
3     div: DB 'Результат: ',0
4 SECTION .text
5     GLOBAL _start
6 _start:
7 ; ---- Вычисление выражения (3+2)*4+5
8     mov ebx,3
9     mov eax,2
10    add eax,ebx
11    mov ecx,4
12    mul ecx
13    add eax,5
14    mov edi,eax
15    mov eax,div
16    call sprintf
17    mov eax,edi
18    call iprintLF
19    call quit
20
```

Рис. 4.47: Изменённый текст программы

46. Создал исполняемый файл и проверил его работу (рис. 4.48).



```
avpereregudov@Study:~/work/arch-pc/lab09$ nasm -f elf task_2.asm
avpereregudov@Study:~/work/arch-pc/lab09$ ld -m elf_i386 -o task_2 task_2.o
avpereregudov@Study:~/work/arch-pc/lab09$ ./task_2
Результат: 25
```

Рис. 4.48: Результат работы программы

5 Выводы

В этой лабораторной работе были приобретены навыки написания программ с использованием подпрограмм, а также были рассмотрены методы отладки при помощи GDB и его основные возможности.

Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016.
URL: <https://www.gnu.org/software/bash/manual/>.
2. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.
6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.