

# **Отчёт по лабораторной работе №9**

**Дисциплина: Архитектура компьютера**

Мария Данииловна Гольцова

# Содержание

1	Цель работы	6
2	Выполнение лабораторной работы	7
3	Самостоятельная работа	22
4	Выводы	27
	Список литературы	28

# Список иллюстраций

2.1	Создание каталога, файла . . . . .	7
2.2	Текст программы 1 . . . . .	8
2.3	Работа программы 1 . . . . .	8
2.4	Изменение программы 1 . . . . .	9
2.5	Работа программы . . . . .	10
2.6	Текст программы 2 . . . . .	11
2.7	Исполняемый файл . . . . .	12
2.8	Загрузка файла в отладчик gdb . . . . .	12
2.9	Проверка работы программы . . . . .	12
2.10	Установка брейкпоинта и запуск программы . . . . .	13
2.11	Дисассимилированный код программы . . . . .	13
2.12	Переключение на отображение команд . . . . .	14
2.13	Режим псевдографики . . . . .	15
2.14	Режим псевдографики . . . . .	15
2.15	Точка останова . . . . .	16
2.16	Адрес предпоследней инструкции . . . . .	16
2.17	Информация о всех установленных точках останова . . . . .	16
2.18	Содержимое регистров . . . . .	17
2.19	Значение переменной 1 . . . . .	17
2.20	Значение переменной 2 . . . . .	17
2.21	Изменение первого символа msg1 . . . . .	17
2.22	Изменение первого символа msg2 . . . . .	18
2.23	Вывод значений регистра edx . . . . .	18
2.24	Изменение регистра ebx . . . . .	19
2.25	Копирование . . . . .	19
2.26	Создание исполняемого файла . . . . .	19
2.27	Загрузка в отладчик . . . . .	20
2.28	Установка точки останова . . . . .	20
2.29	Адрес вершины стека . . . . .	20
2.30	Позиции стека . . . . .	21
3.1	Новый файл . . . . .	22
3.2	Преобразованная программа . . . . .	23
3.3	Результат . . . . .	24
3.4	Новый файл . . . . .	24
3.5	Текст программы . . . . .	24
3.6	Запуск . . . . .	25

3.7	Дисассимилированный код, значения регистров . . . . .	25
3.8	Измененная программа . . . . .	26
3.9	Результат . . . . .	26

## **Список таблиц**

# 1 Цель работы

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

## 2 Выполнение лабораторной работы

Создала каталог для выполнения лабораторной работы №9, перешла в него и создала файл (рис. 2.1).

```
mdgoljcova@dk8n77 ~ $ mkdir ~/work/arch-pc/lab09
mdgoljcova@dk8n77 ~ $ cd ~/work/arch-pc/lab09
mdgoljcova@dk8n77 ~/work/arch-pc/lab09 $ touch lab09-1.asm
mdgoljcova@dk8n77 ~/work/arch-pc/lab09 $
```

Рис. 2.1: Создание каталога, файла

Ввела в файл lab09-1.asm текст программы из листинга 9.1 (рис. 2.2).

```

lab09-1.asm      [-M--]  9 L:[ 1+32 33/ 35] *(646 / 707
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите x: ',0
result: DB '2x+7=',0
SECTION .bss
x: RESB 80
res: RESB 80
SECTION .text
GLOBAL _start
_start:
;-----
; Основная программа
;-----
mov eax, msg
call sprint
mov ecx, x
mov edx, 80
call sread
mov eax, x
call atoi
call _calcul ; Вызов подпрограммы _calcul
mov eax, result
call sprint
mov eax, [res]
call iprintLF
call quit
;-----
; Подпрограмма вычисления
; выражения "2x+7"
_calcul:
mov ebx, 2
mul ebx
add eax, 7
mov [res], eax
ret ; выход из подпрограммы

```

Рис. 2.2: Текст программы 1

Создала исполняемый файл и проверила его работу (рис. 2.3).

```

mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ nasm -f elf lab09-1.asm
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-1 lab09-1.o
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ ./lab09-1
Введите x: 5
2x+7=17
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $

```

Рис. 2.3: Работа программы 1



Изменила 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. 2.4).

```
lab09-1.asm      [-M--]  3 L:[  1+34  35/ 35
#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 ; Вызов подпрограммы _calcul
mov eax, result
call sprint
mov eax, [res]
call iprintLF
call quit
_calcul:
call _subcalcul
mov ebx, 2
mul ebx
add eax, 7
mov [res], eax
ret ; выход из подпрограммы
_subcalcul:
mov ebx, 3
mul ebx
sub eax, 1
ret
```

Рис. 2.4: Изменение программы 1

Создала исполняемый файл и проверила его работу (рис. 2.5).

```
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ nasm -f elf lab09-1.asm
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-1 lab09-1.o
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ ./lab09-1
Введите x: 5
2(3x-1)+7=35
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $
```

Рис. 2.5: Работа программы

Создала файл lab09-2.asm с текстом программы из листинга 9.2 (рис. 2.6).

```
lab09-2.asm      [-M--  
SECTION .data  
msg1: db "Hello, ",0x0  
msg1Len: equ $ - msg1  
msg2: db "world!",0xa  
msg2Len: equ $ - msg2  
SECTION .text  
global _start  
_start:  
mov eax, 4  
mov ebx, 1  
mov ecx, msg1  
mov edx, msg1Len  
int 0x80  
mov eax, 4  
mov ebx, 1  
mov ecx, msg2  
mov edx, msg2Len  
int 0x80  
mov eax, 1  
mov ebx, 0  
int 0x80
```

Рис. 2.6: Текст программы 2

Получила исполняемый файл (рис. 2.7).

```
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ nasm -f elf -g -l lab09-2.lst lab09-2.asm
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-2 lab09-2.o
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $
```

Рис. 2.7: Исполняемый файл

Загрузила исполняемый файл в отладчик gdb (рис. 2.8).

```
mdgoljcova@dk8n52 ~/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. 2.8: Загрузка файла в отладчик gdb

Проверила работу программы, запустив её в оболочке GDB с помощью команды run (рис. 2.9).

```
(gdb) r
Starting program: /afs/.dk.sci.pfu.edu.ru/home/m/d/mdgoljcova/work/arch-pc/lab09/lab09-2
Hello, world!
[Inferior 1 (process 3908) exited normally]
(gdb)
```

Рис. 2.9: Проверка работы программы

Для более подробного анализа программы установила брейкпоинт на метку \_start, с которой начинается выполнение любой ассемблерной программы, и запустила её (рис. 2.10).

```
(gdb) break _start
Breakpoint 1 at 0x08049000: file lab09-2.asm, line 9.
(gdb) r
Starting program: /afs/.dk.sci.pfu.edu.ru/home/m/d/mdgoljcova/work/arch-pc/lab09/lab09-2
Breakpoint 1, _start () at lab09-2.asm:9
```

Рис. 2.10: Установка брейкпоинта и запуск программы

Посмотрела дисассимилированный код программы с помощью команды `disassemble`, начиная с метки `_start` (рис. 2.11).

```
(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.
(gdb)
```

Рис. 2.11: Дисассимилированный код программы

Переключение на отображение команд с Intelовским синтаксисом (рис. 2.12).

```

(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.
(gdb) █

```

Рис. 2.12: Переключение на отображение команд

Включила режим псевдографики для более удобного анализа программы:  
 layout asm (рис. 2.13).

```
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
0x8049038                add     BYTE PTR [eax],al
0x804903a                add     BYTE PTR [eax],al
0x804903c                add     BYTE PTR [eax],al
0x804903e                add     BYTE PTR [eax],al
0x8049040                add     BYTE PTR [eax],al
0x8049042                add     BYTE PTR [eax],al
0x8049044                add     BYTE PTR [eax],al
0x8049046                add     BYTE PTR [eax],al
0x8049048                add     BYTE PTR [eax],al

native process 3950 In: _start
(gdb) 
```

Рис. 2.13: Режим псевдографики

layout regs (рис. 2.14).

```
[ 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
0x8049020 <_start+32>     mov     ecx,0x804a008
0x8049025 <_start+37>     mov     edx,0x7
0x804902a <_start+42>     int      0x80
0x804902c <_start+44>     mov     eax,0x1

native process 3950 In: _start
(gdb) layout regs
(gdb) 
```

Рис. 2.14: Режим псевдографики

Проверила точку останова по имени метки (рис. 2.15).

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

Рис. 2.15: Точка останова

Определила адрес предпоследней инструкции (`mov ebx,0x0`) и установила точку останова (рис. 2.16).

```
(gdb) break *0x8049031
Breakpoint 2 at 0x8049031: file lab09-2.asm, line 20.
(gdb) █
```

Рис. 2.16: Адрес предпоследней инструкции

Посмотрела информацию о всех установленных точках останова (рис. 2.17).

```
(gdb) i b
Num      Type           Disp Enb Address      What
1        breakpoint     keep y   0x08049000 lab09-2.asm:9
          breakpoint already hit 1 time
2        breakpoint     keep y   0x08049031 lab09-2.asm:20
(gdb) █
```

Рис. 2.17: Информация о всех установленных точках останова

Посмотрела содержимое регистров (рис. 2.18).



```

native process 3950 In: _start
eax      0x0      0
ecx      0x0      0
edx      0x0      0
ebx      0x0      0
esp      0xffffc310 0xffffc310
ebp      0x0      0x0
esi      0x0      0
edi      0x0      0
eip      0x8049000 0x8049000 <_start>
eflags   0x202    [ IF ]
cs       0x23     35
ss       0x2b     43
--Type <RET> for more, q to quit, c to continue without paging--

```

Рис. 2.18: Содержимое регистров

Посмотрела значение переменной msg1 по имени (рис. 2.19).

```

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

```

Рис. 2.19: Значение переменной 1

Посмотрела значение переменной msg2 по адресу (рис. 2.20).

```

(gdb) x/1sb 0x804a008
0x804a008 <msg2>:      "world!\n\034"
(gdb)

```

Рис. 2.20: Значение переменной 2

Изменила первый символ переменной msg1 (рис. 2.21).

```

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

```

Рис. 2.21: Изменение первого символа msg1

Заменяла первый символ во второй переменной msg2 (рис. 2.22).

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

Рис. 2.22: Изменение первого символа msg2

Вывела в различных форматах (в шестнадцатеричном формате, в двоичном формате и в символьном виде) значение регистра edx (рис. 2.23).

```
(gdb) p/s $edx  
$1 = 0  
(gdb) p/t $edx  
$2 = 0  
(gdb) p/x $edx  
$3 = 0x0  
(gdb) █
```

Рис. 2.23: Вывод значений регистра edx

С помощью команды set изменила значение регистра ebx (рис. 2.24).

```

(gdb) set $ebx='2'
(gdb) p/s $ebx
$4 = 50
(gdb) set $ebx=2
(gdb) p/s $ebx
$5 = 2
(gdb)

```

Рис. 2.24: Изменение регистра ebx

Завершила выполнение программы с помощью команды `continue` (сокращенно `c`) или `stepi` (сокращенно `si`) и вышла из GDB с помощью команды `quit` (сокращенно `q`).

Скопировала файл `lab8-2.asm`, созданный при выполнении лабораторной работы №8, с программой выводящей на экран аргументы командной строки (Листинг 8.2) в файл с именем `lab09-3.asm` (рис. 2.25).

```

mdgoljcovae@dk8n52 ~/work/arch-pc/lab09 $ cd
mdgoljcovae@dk8n52 ~ $ cp ~/work/arch-pc/lab08/lab8-2.asm ~/work/arch-pc/lab09/lab09-3.asm
mdgoljcovae@dk8n52 ~ $

```

Рис. 2.25: Копирование

Создала исполняемый файл (рис. 2.26).

```

mdgoljcovae@dk8n52 ~ $ cd ~/work/arch-pc/lab09
mdgoljcovae@dk8n52 ~/work/arch-pc/lab09 $ nasm -f elf -g -l lab09-3.lst lab09-3.asm
mdgoljcovae@dk8n52 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-3 lab09-3.o
mdgoljcovae@dk8n52 ~/work/arch-pc/lab09 $

```

Рис. 2.26: Создание исполняемого файла

Загрузила исполняемый файл в отладчик, указав аргументы (рис. 2.27).

```
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ gdb --args lab09-3 аргумент1 аргумент 2 'аргумент 3'
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 lab09-3...
(gdb) 
```

Рис. 2.27: Загрузка в отладчик

Для начала установлю точку останова перед первой инструкцией в программе и запущу ее (рис. 2.28).

```
(gdb) b _start
Breakpoint 1 at 0x80490e8: file lab09-3.asm, line 8.
(gdb) run
Starting program: /afs/.dk.sci.pfu.edu.ru/home/m/d/mdgoljcova/work/arch-pc/lab09/lab09-3 аргумент1 аргумент 2 аргумент\ 3

Breakpoint 1, _start () at lab09-3.asm:8
8      pop ecx ; Извлекаем из стека в 'ecx' количество
(gdb) 
```

Рис. 2.28: Установка точки останова

Адрес вершины стека храниться в регистре esp, и по этому адресу располагается число, равное количеству аргументов командной строки (включая имя программы) (рис. 2.29).

```
(gdb) x/x $esp
0xffffc2c0: 0x00000005
(gdb) 
```

Рис. 2.29: Адрес вершины стека

Посмотрела остальные позиции стека (рис. 2.30).

```

(gdb) x/s *(void**)(esp + 4)
0xffffc55d:    "/afs/.dk.sci.pfu.edu.ru/home/m/d/mdgoljcova/work/arch-pc/lab09/lab09-3"
(gdb) x/s *(void**)(esp + 8)
0xffffc5a4:    "аргумент1"
(gdb) x/s *(void**)(esp + 12)
0xffffc5b6:    "аргумент"
(gdb) x/s *(void**)(esp + 16)
0xffffc5c7:    "2"
(gdb) x/s *(void**)(esp + 20)
0xffffc5c9:    "аргумент 3"
(gdb) x/s *(void**)(esp + 24)
0x0:    <error: Cannot access memory at address 0x0>
(gdb)

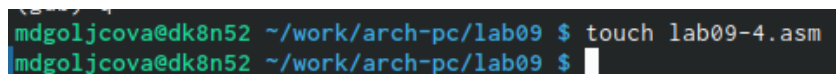
```

Рис. 2.30: Позиции стека

Шаг изменения адреса равен 4, потому что число аргументов равно 4.

### 3 Самостоятельная работа

Создала файл для самостоятельной работы (рис. 3.1).



```
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ touch lab09-4.asm
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $
```

Рис. 3.1: Новый файл

Преобразовала программу из лабораторной работы №8 (Задание №1 для самостоятельной работы), реализовав вычисление значения функции  $\boxtimes(\boxtimes)$  как подпрограмму (рис. 3.2).

```

lab09-4.asm      [----]  3 L:[  1+28
#include 'in_out.asm'
SECTION .data
msg db "Ответ: ",0
SECTION .text
global _start
_start:
pop ecx
pop edx
sub ecx,1
mov esi, 0
next:
cmp ecx,0h
jz _end
pop eax
call atoi
call _calcul
add esi,eax
loop next
_end:
mov eax, msg
call sprint
mov eax, esi
call iprintLF
call quit
_calcul:
mov ebx,2
mul ebx
add eax,15
ret

```

Рис. 3.2: Преобразованная программа

Получила верный ответ (рис. 3.3).

```
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ nasm -f elf -g -l lab09-4.lst lab09-4.asm
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-4 lab09-4.o
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ ./lab09-4 1 2 3 4
Ответ: 80
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $
```

Рис. 3.3: Результат

Создала файл для второго задания самостоятельной работы (рис. 3.4).

```
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $ touch lab09-5.asm
mdgoljcova@dk8n52 ~/work/arch-pc/lab09 $
```

Рис. 3.4: Новый файл

Ввела программу из листинга 9.3 (рис. 3.5).

```
lab09-5.asm [----] 11 L: [ 1+13 14/ 20
#include 'in_out.asm'
SECTION .data
div: DB 'Результат: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения (3+2)*4+5
mov ebx,3
mov eax,2
add ebx,eax
mov ecx,4
mul ecx
add ebx,5
mov edi,ebx
; ---- Вывод результата на экран
mov eax,div
call sprint
mov eax,edi
call iprintLF
call quit
```

Рис. 3.5: Текст программы

Попробовала запустить программу (рис. 3.6).



```

mdgoljcovadk8n52 ~/work/arch-pc/lab09 $ nasm -f elf -g -l lab09-5.lst lab09-5.asm
mdgoljcovadk8n52 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-5 lab09-5.o
mdgoljcovadk8n52 ~/work/arch-pc/lab09 $ gdb lab09-5
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 lab09-5...
(gdb)

```

Рис. 3.6: Запуск

Просмотрела дисассимилированный код программы, поставила точку останова перед прибавлением 5 и открыла значения регистров на данном этапе (рис. 3.7).

```

(gdb) disassemble _start
Dump of assembler code for function _start:
0x080490e8 <+0>:    mov     $0x3,%ebx
0x080490ed <+5>:    mov     $0x2,%eax
0x080490f2 <+10>:   add     %eax,%ebx
0x080490f4 <+12>:   mov     $0x4,%ecx
0x080490f9 <+17>:   mul     %ecx
0x080490fb <+19>:   add     $0x5,%ebx
0x080490fe <+22>:   mov     %ebx,%edi
0x08049100 <+24>:   mov     $0x804a000,%eax
0x08049105 <+29>:   call    0x804900f <sprint>
0x0804910a <+34>:   mov     %edi,%eax
0x0804910c <+36>:   call    0x8049086 <iprintf>
0x08049111 <+41>:   call    0x80490db <quit>
End of assembler dump.
(gdb) b *0x080490fb
Breakpoint 1 at 0x80490fb: file lab09-5.asm, line 13.
(gdb) run
Starting program: /afs/.dk.sci.pfu.edu.ru/home/m/d/mdgoljcovadk8n52/work/arch-pc/lab09/lab09-5

Breakpoint 1, _start () at lab09-5.asm:13
13      add ebx,5
(gdb) i r
eax            0x8          8
ecx            0x4          4
edx            0x0          0
ebx            0x5          5
esp            0xffffc310    0xffffc310
ebp            0x0          0x0
esi            0x0          0
edi            0x0          0
eip            0x80490fb      0x80490fb <_start+19>
eflags        0x202          [ IF ]
cs             0x23          35
ss             0x2b          43
ds             0x2b          43
es             0x2b          43
fs             0x0          0
gs             0x0          0
(gdb)

```

Рис. 3.7: Дисассимилированный код, значения регистров

Регистр ехх со значением 4 умножается не на еbх, сложенным с еах, а только с еах со значением 2. Меняю (рис. 3.8).

```
lab09-5.asm      [-M--]  9 L:[ 1+19 20/ :
#include 'in_out.asm'
SECTION .data
div: DB 'Результат: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения (3+2)*4+5
mov  eax,3
mov  ebx,2
add  eax,ebx
mov  ecx,4
mul  ecx
add  eax,5
mov  edi,eax
; ---- Вывод результата на экран
mov  eax,div
call sprint
mov  eax,edi
call iprintLF
call quit
```

Рис. 3.8: Измененная программа

Получаю верный ответ (рис. 3.9).

```
mdgoljcovae@dk8n52 ~/work/arch-pc/lab09 $ nasm -f elf -g -l lab09-5.lst lab09-5.asm
mdgoljcovae@dk8n52 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-5 lab09-5.o
mdgoljcovae@dk8n52 ~/work/arch-pc/lab09 $ ./lab09-5
Результат: 25
mdgoljcovae@dk8n52 ~/work/arch-pc/lab09 $
```

Рис. 3.9: Результат

## 4 Выводы

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

## **Список литературы**