

Отчёт по лабораторной работе №9

дисциплина: Архитектура компьютера

Попов Даниил Георгиевич

Содержание

1)Цель работы	5
2)Выполнение лабораторной работы	6
2.1)Создаем каталог для программ ЛБ9 и создаем в нем файл	6
2.2)Открываем файл в Midnight Commander и заполняем его согласно листингу 1. Создаем исполняемый файл и проверяем	7
2.3)Изменяем код в файле и проверяем что изменилось	8
2.4)Создаем файл, заполняем в соответствии с листингом 2 и получаем исходный файл с использованием отладчика gdb	9
2.5)Запускаем программу командой run	10
2.6)Устанавливаем брейкпоинт на метку _start и запускаем программу	10
2.7)Смотрим дисассимилированный код программы с помощью команды disassemble, начиная с метки _start	11
2.8)Переключаем на отображение команд с Intel синтаксисом	12
2.9)Включаем отображение регистров, их значений и результат дисассимилирования программы	13
2.10)Проверяем была ли установлена точка остановки и устанавливаем точку остановки последней инструкции	14
2.11)Смотрим информацию о всех установленных точках остановки	14
2.12)Выполняем 5 команд si	15
2.13)Смотрим значение переменной msg1 и переменной msg2	16
2.14)Меняем первый символ переменной msg1 и msg2	16
2.15)Смотрим значение регистра edx в разных форматах	16
2.16)Изменяем регистр ebx	17
2.17)Прописываем команды для завершения программы и выхода из GBD	17
2.18)Копируем файл lab8-2.asm в файл с именем lab9-3.asm создаем и запускаем в отладчике файл	18
2.19)Установим точку остановки перед первой инструкцией в программе и запустим ее	19
2.20)Смотрите позиции стека по разным адресам	19
3)Задания для самостоятельной работы	20
3.1.2)Открываем файл и пишем код	21
3.1.3)Проверяем	22
4)Выводы	25

Список иллюстраций

1	Создаем	6
2	Заполняем, проверяем	7
3	изменяем	8
4	проверяем	8
5	gdb отладчик	9
6	запускаем	10
7	запускаем программу с брейкпоинтом	10
8	запускаем	11
9	intel синтаксис	12
10	включаем	13
11	проверяем	14
12	смотрим	14
13	si	15
14	si	16
15	меняем	16
16	смотрим	16
17	изменяем	17
18	Прописываем	17
19	Прописываем	18
20	Прописываем	19
21	Прописываем	19
1	код	21
2	проверка	22

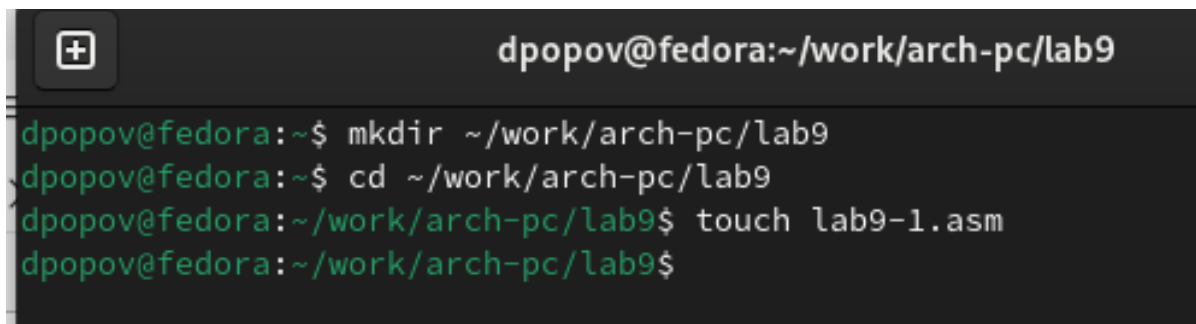
Список таблиц

1)Цель работы

Приобрести навыки написания программ с использованием подпрограмм. Знакомство с методом отладки при помощи GDB и его основными возможностями.

2)Выполнение лабораторной работы

2.1)Создаем каталог для программ ЛБ9 и создаем в нем файл

A terminal window with a dark background. The title bar shows a plus icon in a square and the text 'dpopov@fedora:~/work/arch-pc/lab9'. The terminal contains four lines of text: 'dpopov@fedora:~\$ mkdir ~/work/arch-pc/lab9', 'dpopov@fedora:~\$ cd ~/work/arch-pc/lab9', 'dpopov@fedora:~/work/arch-pc/lab9\$ touch lab9-1.asm', and 'dpopov@fedora:~/work/arch-pc/lab9\$'.

```
dpopov@fedora:~$ mkdir ~/work/arch-pc/lab9
dpopov@fedora:~$ cd ~/work/arch-pc/lab9
dpopov@fedora:~/work/arch-pc/lab9$ touch lab9-1.asm
dpopov@fedora:~/work/arch-pc/lab9$
```

Рис. 1: Создаем

2.2)Открываем файл в Midnight Commander и заполняем его согласно листингу 1. Создаем исполняемый файл и проверяем

```
drogov@fedora:~/work/arch-pc/lab9$ touch lab9-1.asm
drogov@fedora:~/work/arch-pc/lab9$ mc

drogov@fedora:~/work/arch-pc/lab9$ nasm -f elf lab9-1.asm
lab9-1.asm:1: error: unable to open include file `in_out.asm': No such file or d
irectory
drogov@fedora:~/work/arch-pc/lab9$ nasm -f elf lab9-1.asm
drogov@fedora:~/work/arch-pc/lab9$ ld -m elf_i386 -o lab9-1 lab9-1.o
drogov@fedora:~/work/arch-pc/lab9$ ./lab9-1
Введите x: 5
2x+7=17
drogov@fedora:~/work/arch-pc/lab9$
```

Рис. 2: Заполняем, проверяем

2.3)Изменяем код в файле и проверяем что изменилось

```
_calcul:
call _subcalcul
mov ebx,2
mul ebx
add eax,7
mov [res],eax
ret

_subcalcul:
mov ebx,3
mul ebx
sub eax,7
mov [res],eax
ret
```

Рис. 3: изменяем

```
dpopov@fedora:~/work/arch-pc/lab9$ mc
dpopov@fedora:~/work/arch-pc/lab9$ nasm -f elf lab9-1.asm
dpopov@fedora:~/work/arch-pc/lab9$ ld -m elf_i386 -o lab9-1 lab9-1.o
dpopov@fedora:~/work/arch-pc/lab9$ ./lab9-1
Введите x: 5
2(3x-1)+7=35
dpopov@fedora:~/work/arch-pc/lab9$
```

Рис. 4: проверяем

2.4) Создаем файл, заполняем в соответствии с листингом 2 и получаем исходный файл с использованием отладчика gdb

```
2(3x-1)+7=35
dpopov@fedora:~/work/arch-pc/lab9$ touch lab9-2.asm
dpopov@fedora:~/work/arch-pc/lab9$ mc

dpopov@fedora:~/work/arch-pc/lab9$ nasm -f elf -g -l lab9-2.lst lab9-2.asm
dpopov@fedora:~/work/arch-pc/lab9$ ld -m elf_i386 -o lab9-2 lab9-2.o
dpopov@fedora:~/work/arch-pc/lab9$ 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. 5: gdb отладчик

2.5) Запускаем программу командой run

```
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab9-2...
(gdb) run
Starting program: /home/dpopov/work/arch-pc/lab9/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.
Downloading separate debug info for system-supplied DSO at 0xf7ffc000
Hello, world!
[Inferior 1 (process 4591) exited normally]
(gdb)
```

Рис. 6: запускаем

2.6) Устанавливаем брейкпоинт на метку _start и запускаем программу

```
[Inferior 1 (process 4591) exited normally]
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab9-2.asm, line 9.
(gdb) run
Starting program: /home/dpopov/work/arch-pc/lab9/lab9-2

Breakpoint 1, _start () at lab9-2.asm:9
9      mov eax, 4
(gdb)
```

Рис. 7: запускаем программу с брейкпоинтом

2.7) Смотрим дисассемблированный код программы с помощью команды `disassemble`, начиная с метки `_start`

```
(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. 8: запускаем

2.8)Переключаем на отображение команд с Intel

синтаксисом

```
0x08049036 <+54>:    int     $0x80
End of assembler dump.
(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)
```

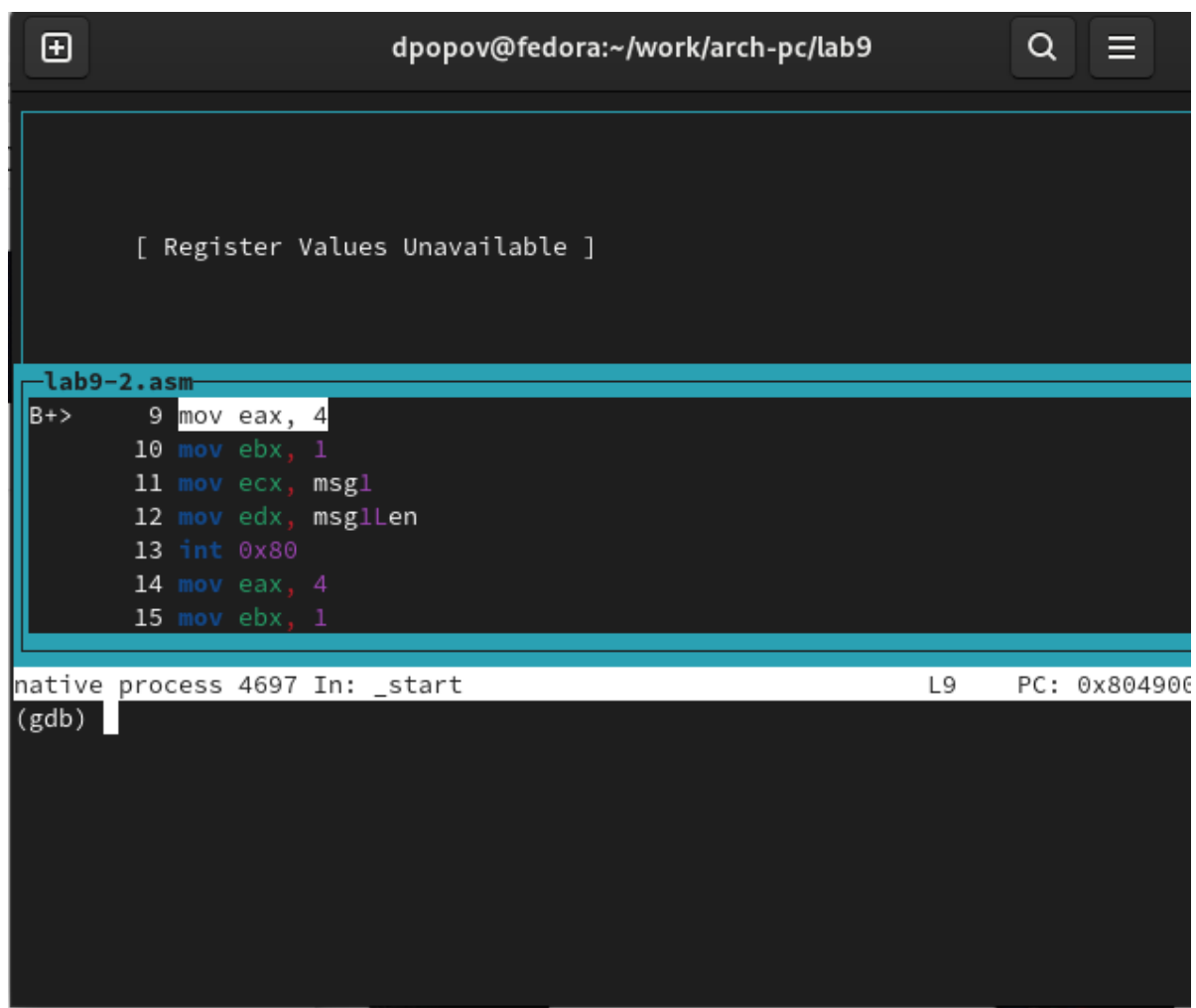
Рис. 9: intel синтаксис

Различия отображения синтаксиса машинных команд в режимах АТТ и Intel:

- 1.Порядок операндов: В АТТ синтаксисе порядок операндов обратный, сначала указывается исходный операнд, а затем - результирующий операнд. В Intel синтаксисе порядок обычно прямой, результирующий операнд указывается первым, а исходный - вторым.
- 2.Разделители: В АТТ синтаксисе разделители операндов - запятые. В Intel синтаксисе разделители могут быть запятые или косые черты (/).
- 3.Префиксы размера операндов: В АТТ синтаксисе размер операнда указывается перед операндом с использованием префиксов, таких как “b” (byte), “w” (word), “l” (long) и “q” (quadword). В Intel синтаксисе размер операнда указывается после операнда с использованием суффиксов, таких как “b”, “w”, “d” и “q”.
- 4.Знак операндов: В АТТ синтаксисе операнды с позитивными значениями предваряются символом “+”. В Intel синтаксисе операнды с позитивными значениями не имеют знака.
- 5.Обозначение адресов: В АТТ синтаксисе адреса указываются в круглых скобках. В Intel синтаксисе адреса указываются без скобок.
- 6.Обозначение регистров: В АТТ синтаксисе обозначение регистра начинается с символа “%”. В Intel синтаксисе обозначение регистра начинается с символа “%”.

обозначение регистра может начинаться с символа “R” или “E” (например, “%eax” или “RAX”).

2.9) Включаем отображение регистров, их значений и результат дисассимилирования программы



The screenshot shows a terminal window with the title bar "dpopov@fedora:~/work/arch-pc/lab9". The main area displays the text "[Register Values Unavailable]". Below this, a file named "lab9-2.asm" is open, showing assembly code:

```
B+> 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
```

At the bottom, the GDB prompt shows "native process 4697 In: _start" and "L9 PC: 0x8049000". The prompt "(gdb)" is visible.

Рис. 10: включаем

2.10)Проверяем была ли установлена точка остановки и устанавливаем точку остановки последней инструкции

```
native process 4697 In: _start L9 PC: 0x8049000
1 breakpoint keep y 0x08049000 lab9-2.asm:9
  breakpoint already hit 1 time
(gdb) break 0x08049000
Function "0x08049000" not defined.
Make breakpoint pending on future shared library load? (y or [n]) n
(gdb) break *0x8049000
Note: breakpoint 1 also set at pc 0x8049000.
Breakpoint 2 at 0x8049000: file lab9-2.asm, line 9.
(gdb)
```

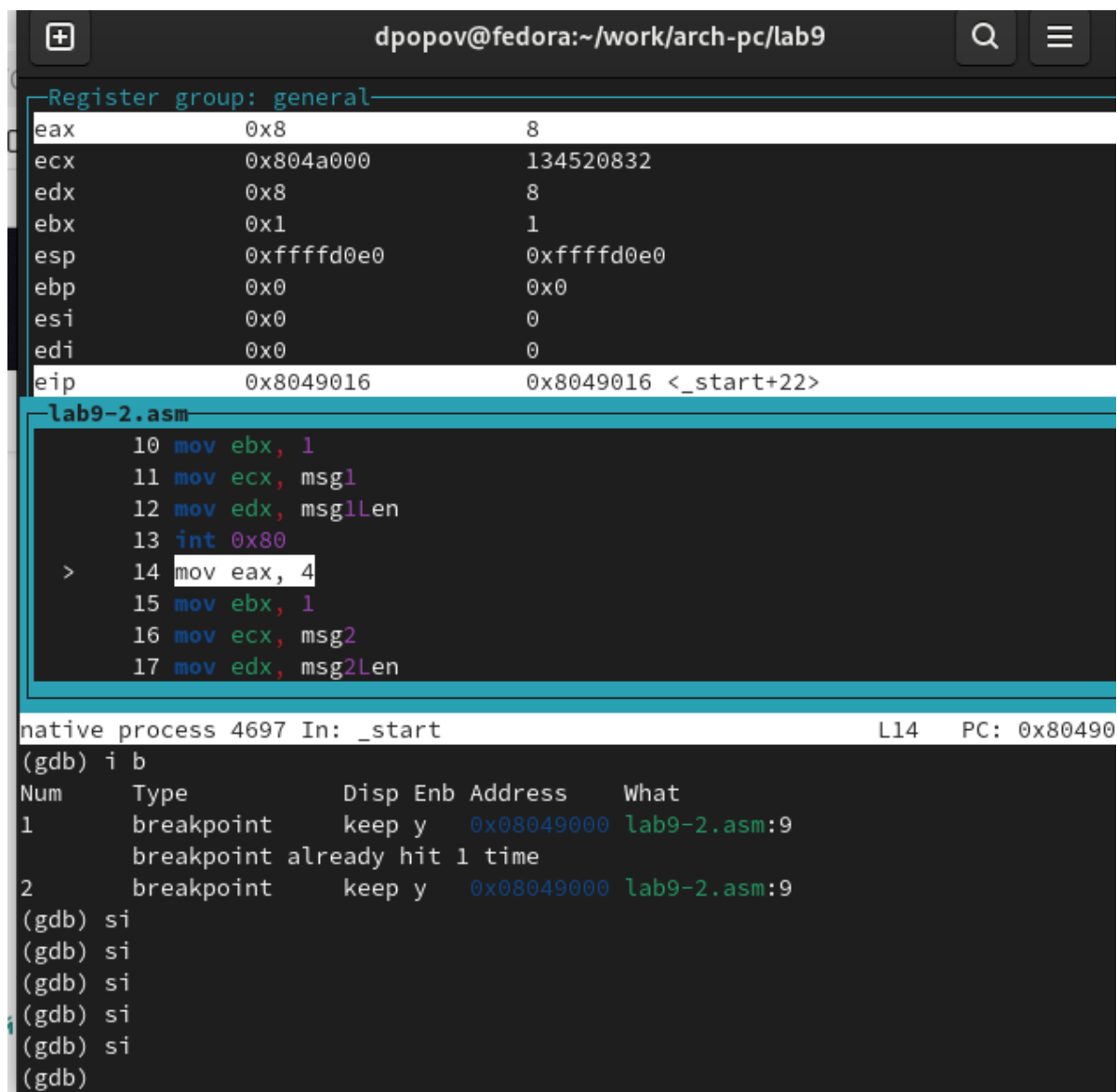
Рис. 11: проверяем

2.11)Смотрим информацию о всех установленных точках остановки

```
(gdb) i b
Num      Type      Disp Enb Address      What
1        breakpoint keep y 0x08049000 lab9-2.asm:9
          breakpoint already hit 1 time
2        breakpoint keep y 0x08049000 lab9-2.asm:9
(gdb) █
```

Рис. 12: смотрим

2.12)Выполняем 5 команд si



The screenshot shows a GDB terminal window with the following content:

```
dropov@fedora:~/work/arch-pc/lab9
```

Register group: general

Register	Value (Hex)	Value (Dec)
eax	0x8	8
ecx	0x804a000	134520832
edx	0x8	8
ebx	0x1	1
esp	0xffffd0e0	0xffffd0e0
ebp	0x0	0x0
esi	0x0	0
edi	0x0	0
eip	0x8049016	0x8049016 <_start+22>

lab9-2.asm

```
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
```

native process 4697 In: _start L14 PC: 0x8049016

```
(gdb) i b
Num    Type           Disp Enb Address      What
1      breakpoint    keep y  0x08049000 lab9-2.asm:9
       breakpoint already hit 1 time
2      breakpoint    keep y  0x08049000 lab9-2.asm:9
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) si
```

Рис. 13: si

Во время выполнения команд менялись регистры: ebx, ecx, edx, eax, eip

2.13) Смотрим значение переменной msg1 и переменной msg2

```
(gdb) si
(gdb) x/1sb & msg1
0x804a000 <msg1>:      "Hello, "
(gdb) x/1sb & msg2
0x804a008 <msg2>:      "world!\n\034"
(gdb) █
```

Рис. 14: si

2.14) Меняем первый символ переменной msg1 и msg2

```
(gdb) set {char}&msg1='h'
(gdb) x/1sb & msg1
0x804a000 <msg1>:      "hello, "
(gdb) set {char}&msg2='L'
(gdb) x/1sb & msg2
0x804a008 <msg2>:      "Lor!d!\n\034"
(gdb) █
```

Рис. 15: меняем

2.15) Смотрим значение регистра edx в разных форматах

```
(gdb) p/t $edx
$1 = 1000
(gdb) p/s $edx
$2 = 8
(gdb) p/x $edx
$3 = 0x8
(gdb) █
```

Рис. 16: смотрим

2.16)Изменяем регистр ebx

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

Рис. 17: изменяем

2.17)Прописываем команды для завершения программы и выхода из GDB

```
(gdb) c  
Continuing.  
World!  
[Inferior 1 (process 4697) exited normally]  
(gdb) quit
```

Рис. 18: Прописываем

2.18) Копируем файл lab8-2.asm в файл с именем lab9-3.asm создаем и запускаем в отладчике файл

```
ddpopov@fedora:~/work/arch-pc/lab9$ cp ~/work/arch-pc/lab8/lab8-2.asm ~/work/arch-pc/lab9/lab9-3.asm
ddpopov@fedora:~/work/arch-pc/lab9$ nasm -f elf -g -l lab9-3.lst lab9-3.asm
ddpopov@fedora:~/work/arch-pc/lab9$ ld -m elf_i386 -o lab0-3 lab9-3.o
ddpopov@fedora:~/work/arch-pc/lab9$ gdb --args lab9-3 2 3 '5'
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"...
lab9-3: Нет такого файла или каталога.
(gdb) gdb --args lab0-3 2 3 '5'
Undefined command: "gdb". Try "help".
(gdb) quit
ddpopov@fedora:~/work/arch-pc/lab9$ gdb --args lab0-3 2 3 '5'
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 lab0-3...
(gdb) █
```

Рис. 19: Прописываем

2.19) Установим точку останова перед первой инструкцией в программе и запустим ее

```
(gdb) b _start
Breakpoint 1 at 0x80490e8: file lab9-3.asm, line 5.
(gdb) run
Starting program: /home/dpopov/work/arch-pc/lab9/lab0-3 2 3 5

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:5
5      pop ecx ; Извлекаем из стека в `ecx` количество
(gdb) x/x $esp
0xffffd0c0:    0x00000004
(gdb)
```

Рис. 20: Прописываем

2.20) Смотрите позиции стека по разным адресам

```
5      pop ecx ; Извлекаем из стека в `ecx` количество
(gdb) x/x $esp
0xffffd0c0:    0x00000004
(gdb) x/s *(void**)($esp + 4)
Argument to arithmetic operation not a number or boolean.
(gdb) x/s *(void**)($esp+4)
0xffffd28b:    "/home/dpopov/work/arch-pc/lab9/lab0-3"
(gdb) x/s *(void**)($esp+8)
0xffffd2b1:    "2"
(gdb) x/s *(void**)($esp+12)
0xffffd2b3:    "3"
(gdb) x/s *(void**)($esp+16)
0xffffd2b5:    "5"
(gdb) x/s *(void**)($esp+20)
0x0:    <error: Cannot access memory at address 0x0>
(gdb)
```

Рис. 21: Прописываем

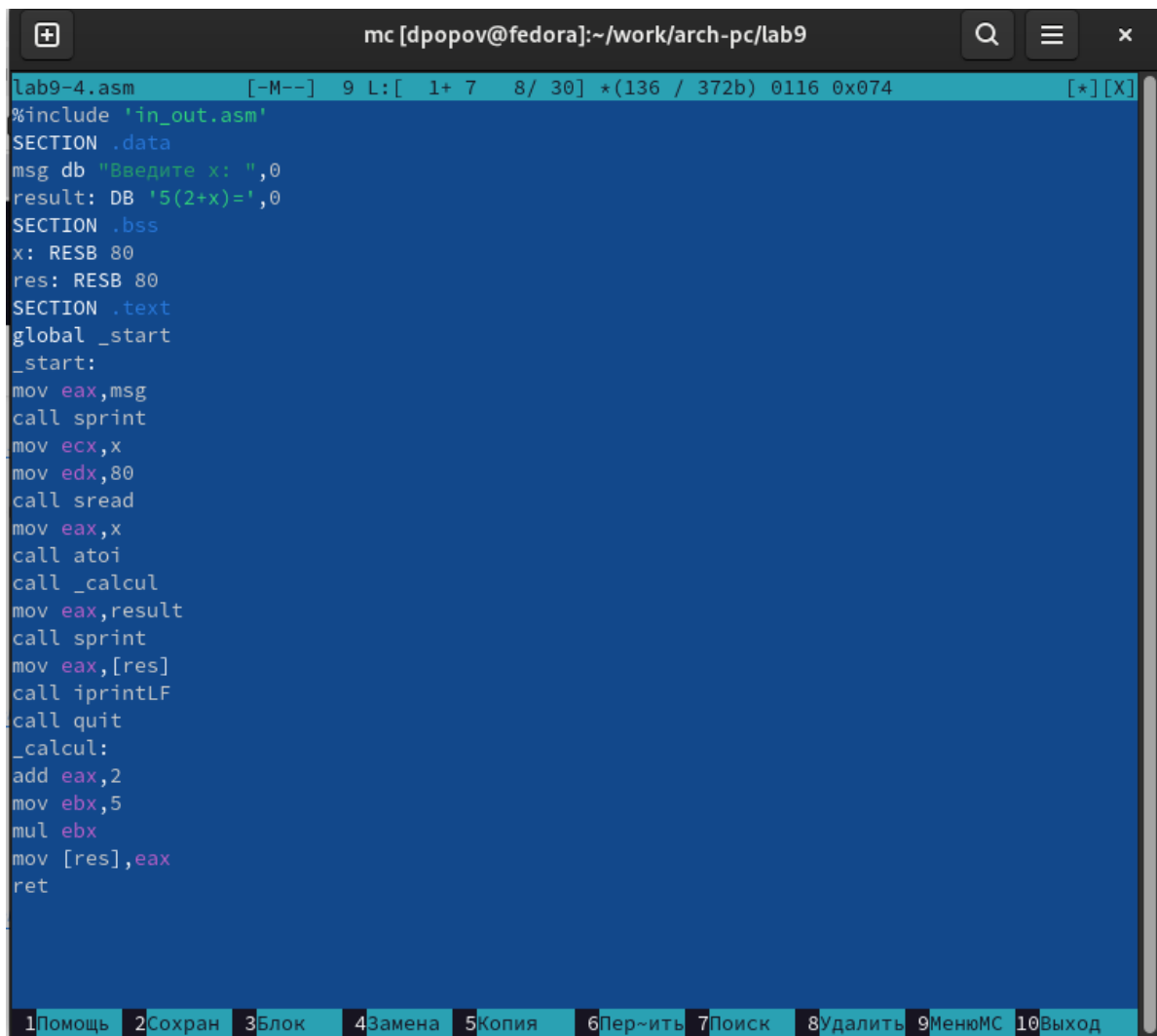
Шаг изменения адреса равен 4 потому что адресная регистрация имеют размерность 32 бита (4 байта)

3)Задания для самостоятельной работы

ВАРИАНТ 10 ##Задание 1 ### 3.1.1)Копируем файл lab8-4.asm в файл с именем lab9-3.asm

```
dpopov@fedora:~/work/arch-pc/lab9$ cp ~/work/arch-pc/lab8/lab8-4.asm ~/work/arch-pc/lab9/lab9-4.asm
dpopov@fedora:~/work/arch-pc/lab9$ mc
dpopov@fedora:~/work/arch-pc/lab9$
```

3.1.2) Открываем файл и пишем код



The screenshot shows the MASM editor window titled "mc [dporov@fedora]:~/work/arch-pc/lab9". The file being edited is "lab9-4.asm". The code is as follows:

```
lab9-4.asm      [-M--]  9 L:[  1+ 7   8/ 30] *(136 / 372b) 0116 0x074  [*] [X]
%include 'in_out.asm'
SECTION .data
msg db "Введите x: ",0
result: DB '5(2+x)=',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:
add eax,2
mov ebx,5
mul ebx
mov [res],eax
ret
```

At the bottom of the window, there is a menu bar with the following options: 1Помощь, 2Сохран, 3Блок, 4Замена, 5Копия, 6Пер~ить, 7Поиск, 8Удалить, 9МенюМС, 10Выход.

Рис. 1: код

3.1.3)Проверяем

```
drogov@fedora:~/work/arch-pc/lab9$ mc
drogov@fedora:~/work/arch-pc/lab9$ nasm -f elf lab9-4.asm
drogov@fedora:~/work/arch-pc/lab9$ ld -m elf_i386 -o lab9-4 lab9-4.o
drogov@fedora:~/work/arch-pc/lab9$ ./lab9-4
Введите x: 5
5(2+x)=35
drogov@fedora:~/work/arch-pc/lab9$
```

Рис. 2: проверка

##Задание 2 ### 3.2.1)Создаем файл и заполняем файл в соответствии с листингом и проверяем

```
drogov@fedora:~/work/arch-pc/lab9$ touch lab9-5.asm
drogov@fedora:~/work/arch-pc/lab9$ mc
drogov@fedora:~/work/arch-pc/lab9$ nasm -f elf lab9-5.asm
drogov@fedora:~/work/arch-pc/lab9$ ld -m elf_i386 -o lab9-5 lab9-5.o
drogov@fedora:~/work/arch-pc/lab9$ ./lab9-5
Результат: 10
```

работу

Работает неправильно ### 3.2.2)Создаем исполняем файл и запускаем его в отладке GDB и смот-

```
drogov@fedora:~/work/arch-pc/lab9$ nasm -f elf -g -l lab9-5.lst lab9-5.asm
drogov@fedora:~/work/arch-pc/lab9$ ld -m elf_i386 -o lab9-5 lab9-5.o
drogov@fedora:~/work/arch-pc/lab9$ gdb lab9-5
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 lab9-5...
(gdb) layout asm
drogov@fedora:~/work/arch-pc/lab9$
```

рим на изменение решистров командой si

```

dpopov@fedora:~/work/arch-pc/lab9

Register group: general
eax      0x2      2
ecx      0x4      4
edx      0x0      0
ebx      0x5      5
esp      0xffffd0e0 0xffffd0e0
ebp      0x0      0x0
esi      0x0      0
edi      0x0      0
eip      0x80490f9 0x80490f9 <_start+17>
eflags   0x206    [ PF IF ]
cs       0x23     35
ss       0x2b     43

B+ 0x80490e8 <_start>    mov    $0x3,%ebx
   0x80490ed <_start+5>  mov    $0x2,%eax
   0x80490f2 <_start+10>  add    %eax,%ebx
   0x80490f4 <_start+12>  mov    $0x4,%ecx
> 0x80490f9 <_start+17>  mul    %ecx
   0x80490fb <_start+19>  add    $0x5,%ebx
   0x80490fe <_start+22>  mov    %ebx,%edi
   0x8049100 <_start+24>  mov    $0x804a000,%eax
   0x8049105 <_start+29>  call   0x804900f <sprint>
   0x804910a <_start+34>  mov    %edi,%eax
   0x804910c <_start+36>  call   0x8049086 <iprintLF>
   0x8049111 <_start+41>  call   0x80490db <quit>

native process 6343 In: _start
(gdb) si
```

3.2.3) Ищем ошибку

```
mc [dpopov@fedora]:~/work
lab9-5.asm [----] 9 L: [ 1+12 13/ 20] *(221
#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
mov eax,div
call sprint
mov eax,edi
call iprintLF
call quit
```

3.2.4)Изменяем код для корректной работы

```
dpopov@fedora:~/work/arch-pc/lab9$ mc
dpopov@fedora:~/work/arch-pc/lab9$ nasm -f elf_i386 lab9-5.asm
nasm: fatal: unrecognised output format `elf_i386' - use -hf for a list
Type nasm -h for help.
dpopov@fedora:~/work/arch-pc/lab9$ nasm -f elf lab9-5.asm
dpopov@fedora:~/work/arch-pc/lab9$ ld -m elf_i386 -o lab9-5 lab9-5.o
dpopov@fedora:~/work/arch-pc/lab9$ ./lab9-5
Результат: 25
dpopov@fedora:~/work/arch-pc/lab9$
```

3.2.4)Проверяем

4)Выводы

Мы познакомились с методом отладки при помощи GBD и его возможности