

Отчёт по лабораторной работе №10

Дисциплина: Архитектура компьютера

Зарипов Евгений

Содержание

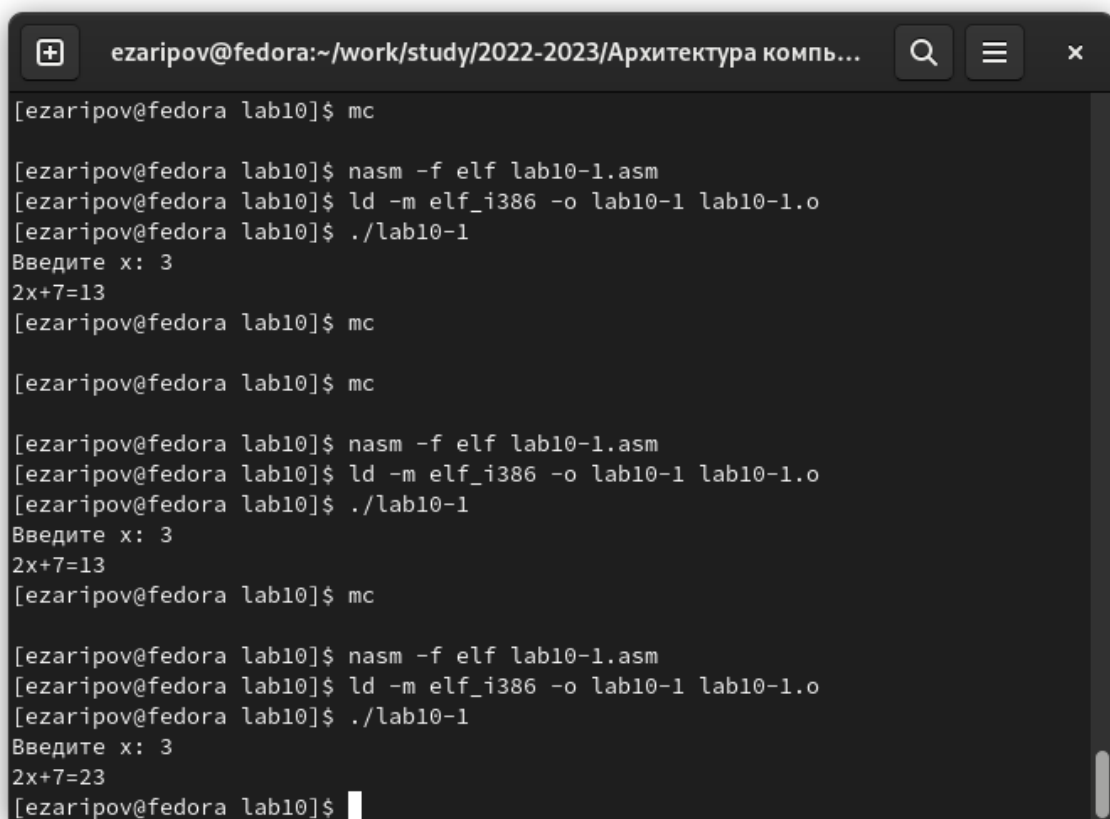
1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Самостоятельная работа	19
4	Выводы	25

1 Цель работы

Научиться работать с отладчиком gdb.

2 Выполнение лабораторной работы

1. С помощью терминала создадим подкаталог, создадим файл lab10-1.asm. Изучим и запишем в него код из листинга, откомпилируем и запустим файл



```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-1.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[ezaripov@fedora lab10]$ ./lab10-1
Введите x: 3
2x+7=13
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-1.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[ezaripov@fedora lab10]$ ./lab10-1
Введите x: 3
2x+7=13
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-1.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[ezaripov@fedora lab10]$ ./lab10-1
Введите x: 3
2x+7=23
[ezaripov@fedora lab10]$
```

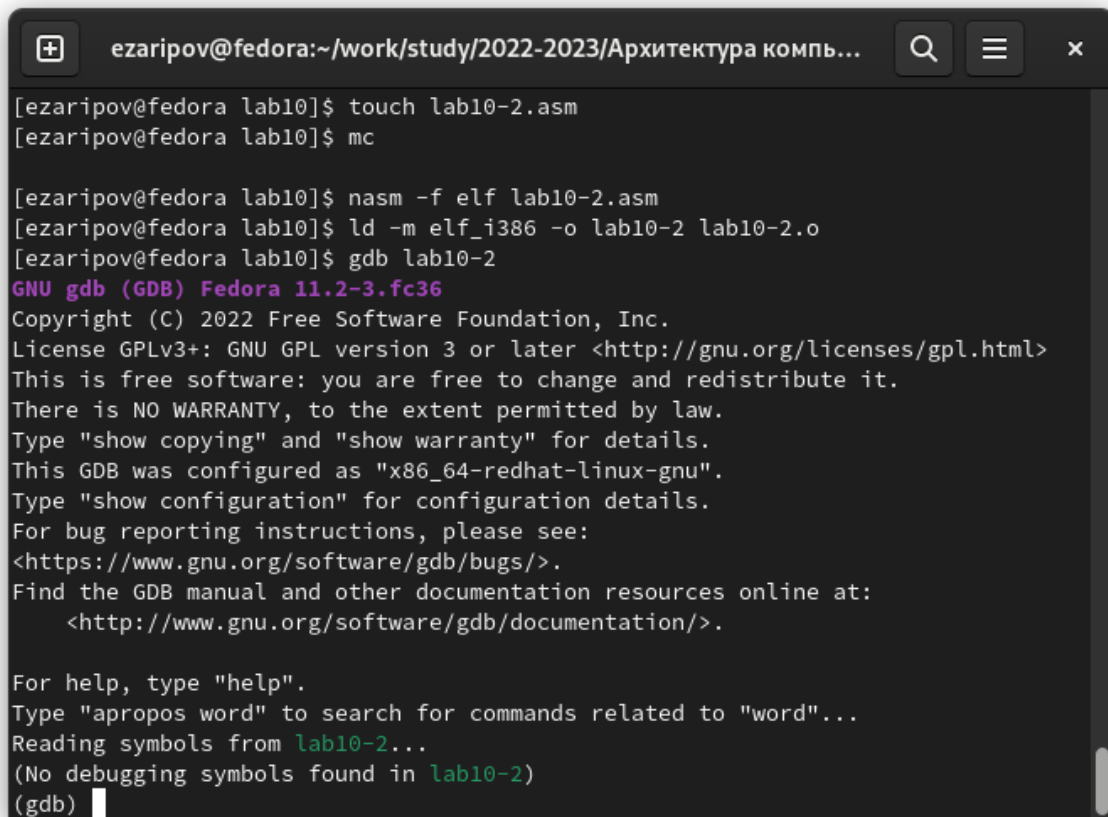
3. Добавим в подпрограмму ещё одну подпрограмму, проверим корректность работы

Открыть lab10-1.asm ~/work/study/202... Сохранить

```
1 %include 'in_out.asm'
2
3 SECTION .data
4     msg: DB 'Введите x: ',0
5     result: DB '2x+7=',0
6
7 SECTION .bss
8     x: RESB 80
9     rez: RESB 80
10
11 SECTION .text
12 GLOBAL _start
13 _start:
14     mov eax, msg
15     call sprint
16     mov ecx, x
17     mov edx, 80
18
19     call sread
20     mov eax,x
21     call atoi
22
23     call _calcul
24
25     mov eax,result
26     call sprint
27     mov eax,[rez]
28     call iprintLF
29     call quit
30
31 _calcul:
32 call _subcalcul
33     mov ebx,2
34     mul ebx
35     add eax,7
36     mov [rez],eax
37     ret
38
39 _subcalcul:
40 mov edx,3
41 mul edx
42 sub eax,1
43 ret
```

Matlab Ширина табуляции: 8 Стр 32, Стлб 16 ВСТ

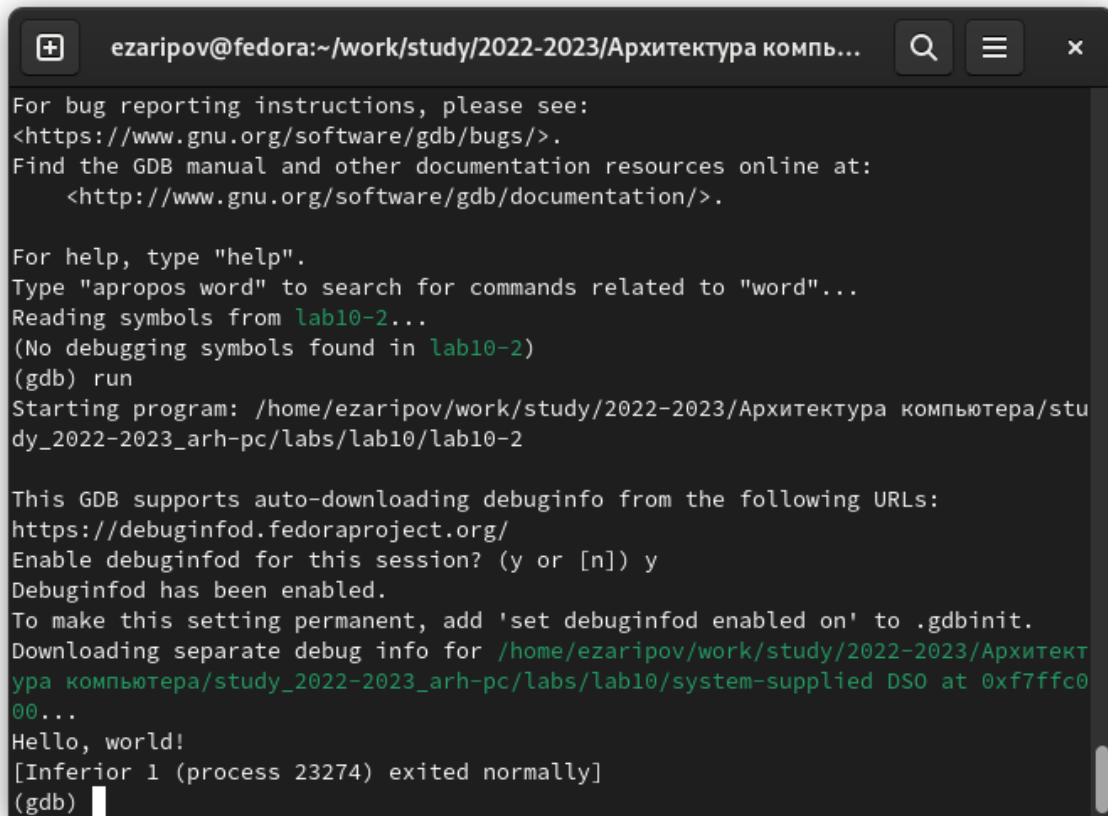
4. Создадим новый файл, запишем в него предложенный код, запустим отладчик и в нем запустим программу



```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
[ezaripov@fedora lab10]$ touch lab10-2.asm
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-2.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-2 lab10-2.o
[ezaripov@fedora lab10]$ gdb lab10-2
GNU gdb (GDB) Fedora 11.2-3.fc36
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-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 lab10-2...
(No debugging symbols found in lab10-2)
(gdb) █
```

A screenshot of a terminal window with a dark background. The title bar at the top shows the user 'ezaripov@fedora' and the current directory path. The terminal displays the output of a GDB session, including instructions for bug reporting, help text, and the execution of a program that prints 'Hello, world!'. The session ends with a normal exit of the inferior process.

```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
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 lab10-2...
(No debugging symbols found in lab10-2)
(gdb) run
Starting program: /home/ezaripov/work/study/2022-2023/Архитектура компьютера/stu
dy_2022-2023_arh-pc/labs/lab10/lab10-2

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 /home/ezaripov/work/study/2022-2023/Архитект
юра компьютера/study_2022-2023_arh-pc/labs/lab10/system-supplied DSO at 0xf7ffc0
00...
Hello, world!
[Inferior 1 (process 23274) exited normally]
(gdb) 
```

5. Установим брейкпоинт

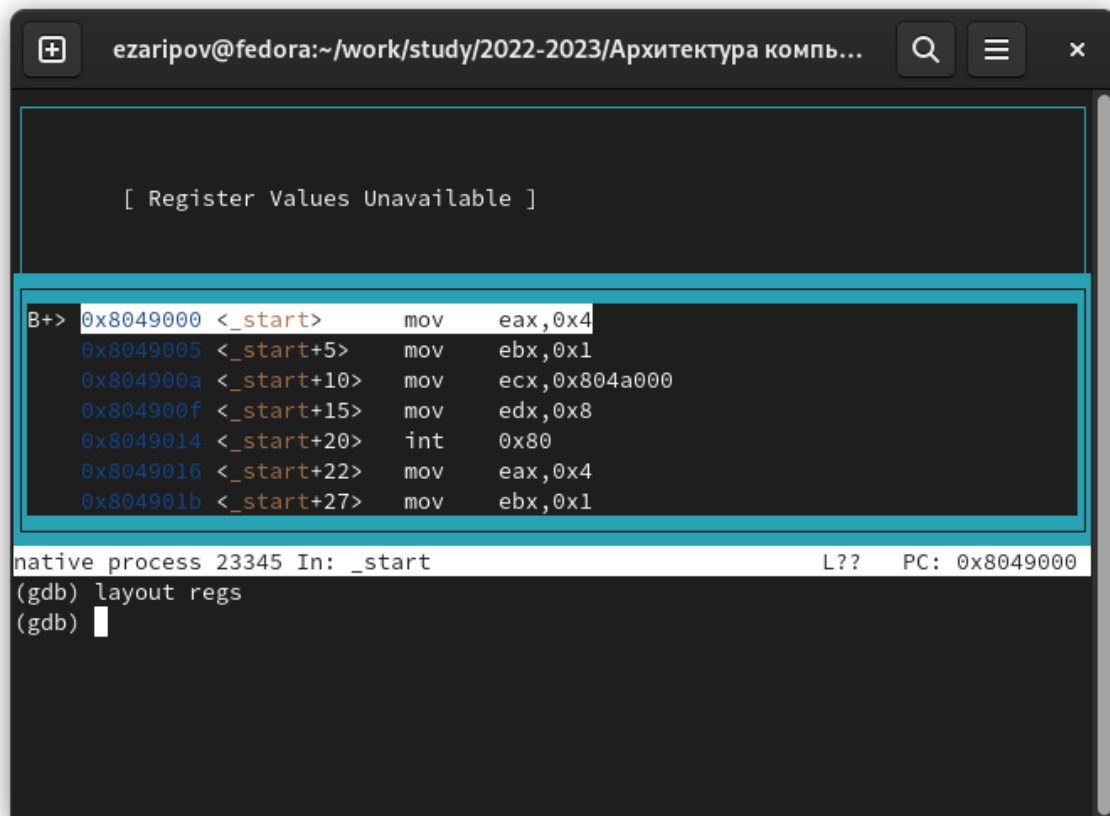
```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
(gdb) run
Starting program: /home/ezaripov/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/labs/lab10/lab10-2

Breakpoint 1, 0x08049000 in _start ()
(gdb) disassemble _start
Undefined command: "disassemble". Try "help".
(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) █
```

6. Рассмотрим отличия между синтаксисами. Ячейки памяти находятся с разных сторон от значений в них и в АТТ добавляются символы \$ и %


```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
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) 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) █
```

7. Выведем режимы псевдографики, по началу layout regs будет пустой



The screenshot shows a GDB terminal window with the title bar "ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...". The window contains the following text:

```
[ 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 23345 In: _start L?? PC: 0x8049000
(gdb) layout regs
(gdb) █

8. Добавим точки останова

```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь... [ 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 23345 In: _start L?? PC: 0x8049000
(gdb) break *0x8049031
Breakpoint 2 at 0x8049031
(gdb) i b
Num Type Disp Enb Address What
1 breakpoint keep y 0x08049000 <_start>
breakpoint already hit 1 time
2 breakpoint keep y 0x08049031 <_start+49>
(gdb)
```

9. С помощью команды `i r` посмотрим содержимое регистров

The screenshot shows a debugger window with the title bar "ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...". The main area displays assembly code with addresses and instructions. A cyan box highlights the first seven lines of code. Below the code, a status bar shows "native process 23345 In: _start" and "PC: 0x8049000". A register window below that shows the values of registers: eax (0x0), ecx (0x0), edx (0x0), ebx (0x0), esp (0xffffd100), ebp (0x0), and esi (0x0). At the bottom, a prompt says "--Type <RET> for more, q to quit, c to continue without paging--".

```
[ Register Values Unavailable ]
```

Address	Instruction
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 23345 In: _start L?? PC: 0x8049000

Register	Value
eax	0x0
ecx	0x0
edx	0x0
ebx	0x0
esp	0xffffd100
ebp	0x0
esi	0x0

--Type <RET> for more, q to quit, c to continue without paging--

10. Теперь поменяем значение в 1 регистре на другое

```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь... [ 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 23345 In: _start L?? PC: 0x8049000
esi 0x0 0
--Type <RET> for more, q to quit, c to continue without paging--qQuit
(gdb) x/1sb &msg1
0x804a000: "Hello, "
(gdb) set {char}&msg1='h'
(gdb) x/1sb &msg1
0x804a000: "hello, "
(gdb) 
```

The screenshot shows a GDB terminal window with the title bar "ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...". The main area displays assembly code for a function starting at address 0x8049000. A cyan box highlights the assembly instructions from 0x8049000 to 0x804901b. Below the assembly, the GDB prompt shows the current state: "native process 23345 In: _start" and "PC: 0x8049000". The user enters the command "set {char}&msg2='???'", which results in an error: "No symbol '???' in current context." The user then enters "x/1sb &msg2", which shows the memory content at 0x804a008 as "world!\n". The user then enters "set {char}&msg2='?'", and the next "x/1sb &msg2" command shows the memory content as "?orld!\n".

```
[ 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 23345 In: _start                                L??  PC: 0x8049000
(gdb) set {char}&msg2='???'
No symbol "???" in current context.
(gdb) x/1sb &msg2
0x804a008:      "world!\n"
(gdb) set {char}&msg2='?'
(gdb) x/1sb &msg2
0x804a008:      "?orld!\n"
(gdb) 
```

11. Воспользуемся функцией (set) и поменяем значение

The screenshot shows a GDB terminal window with the title bar "ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...". The window is divided into two main sections. The top section, titled "Register group: general", displays the values of several registers: `eax` (0x0), `ecx` (0x0), `edx` (0x0), `ebx` (0x32), and `esp` (0xffffd100). The bottom section shows a list of assembly instructions with their addresses and disassembled forms: `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`, and `0x804901b <_start+27> mov ebx,0x1`. Below the assembly list, the status bar indicates "native process 23345 In: _start" and "PC: 0x8049000". The GDB command prompt shows a series of commands: `(gdb) set &ebx='2'`, `(gdb) p/s $ebx`, `(gdb) set $ebx='2'`, and `(gdb) p/s $ebx`, with corresponding outputs: `$3 = 0` and `$4 = 50`.

```
Register group: general
eax      0x0      0
ecx      0x0      0
edx      0x0      0
ebx      0x32     50
esp      0xffffd100 0xffffd100

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 23345 In: _start L?? PC: 0x8049000
(gdb) set &ebx='2'
No symbol "ebx" in current context.
(gdb) p/s $ebx
$3 = 0
(gdb) set $ebx='2'
(gdb) p/s $ebx
$4 = 50
(gdb)
```

12. Запустим программу из 9 лабораторной, установим брейкпоинт и изучим, что лежит в стеке. Шаг равен 4, потому что в 1 ячейке стека 4 байта информации

```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
Type "apropos word" to search for commands related to "word"...
lab10-3: Нет такого файла или каталога.
(gdb) q
[ezaripov@fedora lab10]$ nasm -f elf lab10-3.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-3 lab10-3.o
[ezaripov@fedora lab10]$ gdb --args lab10-3 аргумент1 аргумент 2 'аргумент 3'
GNU gdb (GDB) Fedora 11.2-3.fc36
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-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 lab10-3...
(No debugging symbols found in lab10-3)
(gdb) █
```



```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
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 lab10-3...
(No debugging symbols found in lab10-3)
(gdb) b _start
Breakpoint 1 at 0x80490e8
(gdb) run
Starting program: /home/ezaripov/work/study/2022-2023/Архитектура компьютера/stu
dy_2022-2023_arh-pc/labs/lab10/lab10-3 аргумент1 аргумент 2 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, 0x080490e8 in _start ()
(gdb) 
```

```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
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, 0x080490e8 in _start ()
(gdb) x/x $esp
0xffffd050: 0x00000005
(gdb) x/s *(void**)( $esp + 4)
0xffffd20c: "/home/ezaripov/work/study/2022-2023/Архитектура компьютера/stud
y_2022-2023_arh-pc/labs/lab10/lab10-3"
(gdb) x/s *(void**)( $esp + 8)
0xffffd286: "аргумент1"
(gdb) x/s *(void**)( $esp + 12)
0xffffd298: "аргумент"
(gdb) x/s *(void**)( $esp + 16)
0xffffd2a9: "2"
(gdb) x/s *(void**)( $esp + 20)
0xffffd2ab: "аргумент 3"
(gdb) x/s *(void**)( $esp + 24)
0x0: <error: Cannot access memory at address 0x0>
(gdb) 
```

3 Самостоятельная работа

1. Скопируем файл и изменим код

Открыть  lab10-4.asm  Сохранить  x

```
1 %include 'in_out.asm'
2
3 SECTION .data
4     msg1 db " Функция: f(x)=4x-3", 0
5     msg2 db "Результат: ", 0
6
7 SECTION .text
8 global _start
9 _start:
10     mov eax,msg1
11     call sprintf
12
13     pop ecx
14     pop edx
15     sub ecx,1
16     mov esi,0
17 next:
18     cmp ecx,0h
19     jz _end
20     pop eax
21     call atoi
22     call subcalcul
23     add esi,eax
24     loop next
25 _end:
26     mov eax, msg2
27     call sprintf
28     mov eax, esi
29     call iprintLF
30     call quit
31
32 subcalcul:
33     mov ebx, 4
34     mul ebx
35     add eax, -3
36     ret
```

Matlab  Ширина табуляции: 8  Стр 35, Стлб 16  ВСТ

```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
(gdb) x/s *(void**)(esp + 24)
0x0: <error: Cannot access memory at address 0x0>
(gdb) q
A debugging session is active.

    Inferior 1 [process 24109] will be killed.

Quit anyway? (y or n) y
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab09]$ nasm -f elf lab10-4.asm
nasm: fatal: unable to open input file `lab10-4.asm' No such file or directory
[ezaripov@fedora lab09]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-4.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-4 lab10-4.o
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-4.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-4 lab10-4.o
[ezaripov@fedora lab10]$ ./lab10-4 1 2 3 4
Функция: f(x)=4x-3
Результат: 28
[ezaripov@fedora lab10]$
```

2. Предложенный код выводит ошибку, с помощью gdb и функций X/NFU посмотрим содержание регистра умножения, ещё надо поставить на нем брэйкпоинт, заметим, что в нем изменяется еах, а суммируем мы с ебх и выводим значение в ебх, поэтому заменим в суммирование ебх на еах и получим правильный ответ 25.

```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
Quit anyway? (y or n) y
[ezaripov@fedora lab10]$ mc

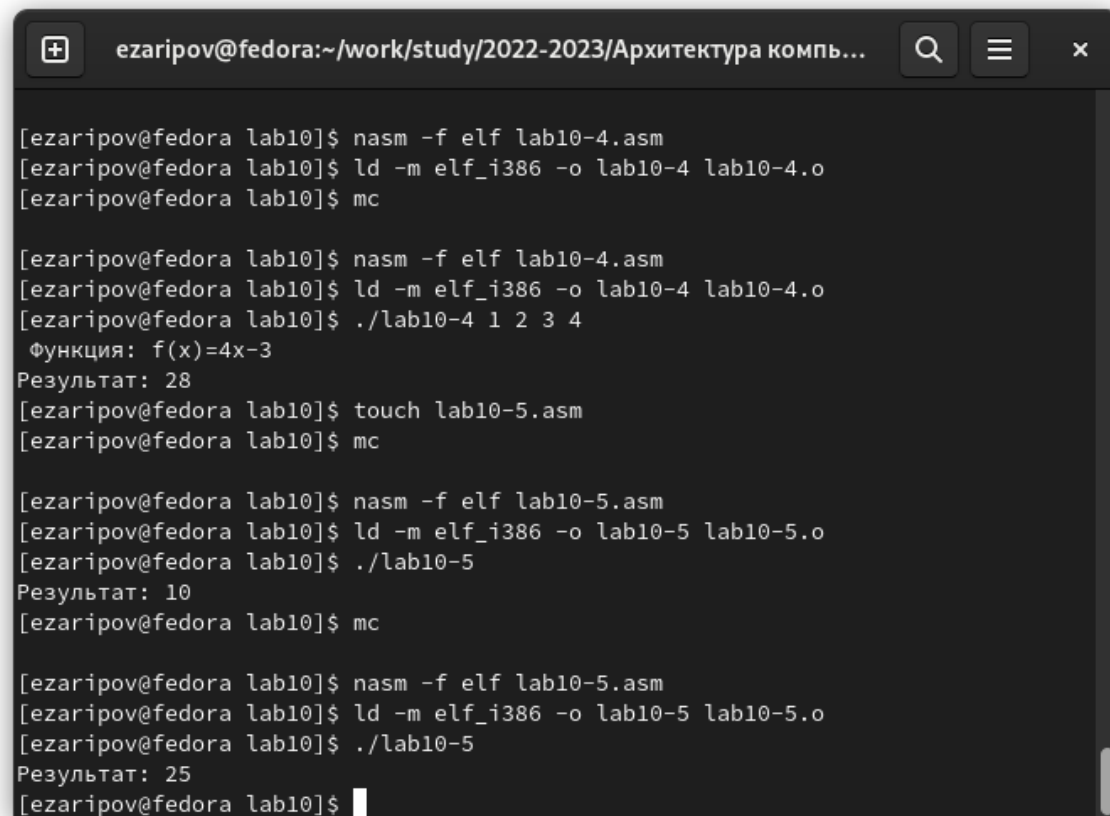
[ezaripov@fedora lab09]$ nasm -f elf lab10-4.asm
nasm: fatal: unable to open input file `lab10-4.asm' No such file or directory
[ezaripov@fedora lab09]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-4.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-4 lab10-4.o
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-4.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-4 lab10-4.o
[ezaripov@fedora lab10]$ ./lab10-4 1 2 3 4
Функция:  $f(x)=4x-3$ 
Результат: 28
[ezaripov@fedora lab10]$ touch lab10-5.asm
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-5.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-5 lab10-5.o
[ezaripov@fedora lab10]$ ./lab10-5
Результат: 10
[ezaripov@fedora lab10]$
```

Рис. 3.1: Неправильный вывод:



```
ezaripov@fedora:~/work/study/2022-2023/Архитектура компь...
[ezaripov@fedora lab10]$ nasm -f elf lab10-4.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-4 lab10-4.o
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-4.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-4 lab10-4.o
[ezaripov@fedora lab10]$ ./lab10-4 1 2 3 4
Функция:  $f(x)=4x-3$ 
Результат: 28
[ezaripov@fedora lab10]$ touch lab10-5.asm
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-5.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-5 lab10-5.o
[ezaripov@fedora lab10]$ ./lab10-5
Результат: 10
[ezaripov@fedora lab10]$ mc

[ezaripov@fedora lab10]$ nasm -f elf lab10-5.asm
[ezaripov@fedora lab10]$ ld -m elf_i386 -o lab10-5 lab10-5.o
[ezaripov@fedora lab10]$ ./lab10-5
Результат: 25
[ezaripov@fedora lab10]$
```

Рис. 3.2: Исправленный вывод

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

Matlab ▾ Ширина табуляции: 8 ▾ Стр 20, Стлб 10 ▾ ВСТ

Рис. 3.3: Исправленный код

4 Выводы

В данной работе мы познакомились с отладчиком и с помощью него научились изменять программу.