

Отчёта по лабораторной работе 10

Отладка

Лев Евгеньевич Гельбарт

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

```
[legeljbart@fedora lab10]$ gedit lab10-1.asm
[legeljbart@fedora lab10]$ nasm -f elf lab10-1.asm
lab10-1.asm:27: error: symbol 'res' not defined
lab10-1.asm:36: error: symbol 'rez' not defined
[legeljbart@fedora lab10]$ gedit lab10-1.asm
[legeljbart@fedora lab10]$ nasm -f elf lab10-1.asm
[legeljbart@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[legeljbart@fedora lab10]$ ./lab10-1
Введите x: 2
2x+7=11
[legeljbart@fedora lab10]$ gedit lab10-1.asm
[legeljbart@fedora lab10]$ nasm -f elf lab10-1.asm
[legeljbart@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[legeljbart@fedora lab10]$ ./lab10-1
Введите x: 2
2x+7=5
[legeljbart@fedora lab10]$ gedit lab10-1.asm
[legeljbart@fedora lab10]$ nasm -f elf lab10-1.asm
[legeljbart@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[legeljbart@fedora lab10]$ ./lab10-1
Введите x: 2
2x+7=17
[legeljbart@fedora lab10]$ touch lab10-2.asm
[legeljbart@fedora lab10]$ gedit lab10-2.asm
[legeljbart@fedora lab10]$ nasm -f elf -g -l lab10-2.lst lab10-2.asm
[legeljbart@fedora lab10]$ ld -m elf_i386 -o lab10-2 lab10-2.o
[legeljbart@fedora lab10]$ gdb lab10-2
GNU gdb (GDB) Fedora 12.1-2.fc36
```

Рис. 2.1: Терминал

Напишем программу по шаблону, увидим что она работает. Затем отредактируем код с созданием дополнительной подпрограммы (текст приложен), видим, что она сработала со второго раза (рис. 2.1).

```

(gdb) r
Starting program: /home/legeljbart/work/arch-pc/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]) n
Debuginfod has been disabled.
To make this setting permanent, add 'set debuginfod enabled off' to .gdbinit.
Hello, world!
[Inferior 1 (process 4136) exited normally]
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab10-2.asm, line 12.
(gdb) r
Starting program: /home/legeljbart/work/arch-pc/lab10/lab10-2

Breakpoint 1, _start () at lab10-2.asm:12
12      mov eax, 4
(gdb)

```

Рис. 2.2: Терминал

Запустим gdb, запустим программу, убедимся в работе. Поставим брейкпоинт и запустим после этого. Видим, что теперь программа сразу заканчивается (рис. 2.2).

```

=> 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) 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. 2.3: Терминал

Просмотрим дисассимилированный код сначала в АТТ синтаксисе, затем в Интеле.(рис. 2.3).

```
(gdb) layout regs
(gdb) i b
Num      Type           Disp Enb Address      What
1        breakpoint     keep y  0x08049000  lab10-2.asm:12
breakpoint already hit 1 time
(gdb)
```

Рис. 2.4: Терминал

Запустим layout, изучим брейкпоинты (рис. 2.4).

```
Num      Type           Disp Enb Address      What
1        breakpoint     keep y  0x08049000  lab10-2.asm:12
breakpoint already hit 1 time
(gdb) b *0x8049031
Breakpoint 2 at 0x8049031: file lab10-2.asm, line 25.
(gdb) i b
Num      Type           Disp Enb Address      What
1        breakpoint     keep y  0x08049000  lab10-2.asm:12
breakpoint already hit 1 time
2        breakpoint     keep y  0x08049031  lab10-2.asm:25
(gdb)
```

Рис. 2.5: Терминал

Поставим брейкпоинт на 0x8049031 (рис. 2.5).

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

Рис. 2.6: Терминал

Поиграем с переменной, заменим Н на h в msg1 (рис. 2.6).

```
0x804a000 <msg1>:      "hello, "
(gdb) set {char}0x804a008='L'
(gdb) set {char}0x804a00b=' '
(gdb) x/1sb &msg2
0x804a008 <msg2>:      "Lor d!\n\034"
(gdb)
```

Рис. 2.7: Терминал

Так же поработаем и с переменной msg2 (рис. 2.7).

```
(gdb) set $ebx='2'
(gdb) p/s $ebx
$1 = 50
(gdb) set $ebx=2
(gdb) p/s $ebx
$2 = 2
(gdb)
```

Рис. 2.8: Терминал

Изменим и выведем значения регистра `ebx`, сначала это 2 как строка/символ, затем это уже число (рис. 2.8).

```
[legeljbart@fedora lab10]$ cp ~/work/arch-pc/lab09/lab9-2.asm ~/work/arch-pc/lab10/lab10-3.asm
[legeljbart@fedora lab10]$ nasm -f elf -g -l lab10-3.lst lab10-3.asm
[legeljbart@fedora lab10]$ ld -m elf_i386 -o lab10-3 lab10-3.o
[legeljbart@fedora lab10]$ gdb --args lab10-3
GNU gdb (GDB) Fedora 12.1-2.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...
(gdb) b _start
Breakpoint 1 at 0x80490e8: file lab10-3.asm, line 6.
(gdb) r
Starting program: /home/legeljbart/work/arch-pc/lab10/lab10-3
```

Рис. 2.9: Терминал

Скопируем написанный ранее код в 10-3 (рис. 2.9).

```
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 lab10-3.asm:6
6      pop ecx
(gdb) x/x $esp
0xffffd1a0: 0x00000005
(gdb)
```

Рис. 2.10: Терминал

Загрузим файл в отладчик указав аргументы, как указано и увидим, что аргументов 5 (рис. 2.10).

```

(gdb) b _start
Breakpoint 1 at 0x80490e8: file lab10-3.asm, line 6.
(gdb) r
Starting program: /home/legeljbart/work/arch-pc/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, _start () at lab10-3.asm:6
6      pop ecx
(gdb) x/x $esp
0xffffd1b0: 0x00000005
(gdb) x/s *(void**)(esp + 4)
0xffffd367: "/home/legeljbart/work/arch-pc/lab10/lab10-3"
(gdb) x/s *(void**)(esp + 8)
0xffffd393: "аргумент1"
(gdb) x/s *(void**)(esp + 12)
0xffffd3a5: "аргумент"
(gdb) x/s *(void**)(esp + 16)
0xffffd3b0: "2"
(gdb) x/s *(void**)(esp + 20)
0xffffd3b8: "аргумент 3"
(gdb) x/s *(void**)(esp + 24)
0x0: <error: Cannot access memory at address 0x0>
(gdb)

```

Рис. 2.11: Терминал

Посмотрим позиции стека, с шагом 4, так как столько занимает ссылка на адрес следующего элемента, видим что аргументы таковы, какими им и следует быть (рис. 2.11).

```

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

```

Рис. 2.12: Код

Представляю Вашему вниманию исправленный и функционирующий код для самостоятельной работы, процесс отладки занимает слишком много скриншотов, поэтому я счел допустимым его опустить. Ошибка была в регистре ebx, он использовался вместо eax во многих строках, что привело к неправильному выполнению программы умножения (рис. 2.12).

```
[legeljbart@fedora lab10]$ gedit lab10-3.asm
[legeljbart@fedora lab10]$ nasm -f elf -g -l lab10-3.lst lab10-3.asm
[legeljbart@fedora lab10]$ ld -m elf_i386 -o lab10-3 lab10-3.o
[legeljbart@fedora lab10]$ ./lab10-3
bash: ./lab10-3: Нет такого файла или каталога
[legeljbart@fedora lab10]$ ./lab10-1
Введите x: 2
2x+7=17
[legeljbart@fedora lab10]$ ./lab10-3
Rezult: 25
[legeljbart@fedora lab10]$
```

Рис. 2.13: Терминал

Видим что программа и правда работает.

3 Выводы

Были получены навыки написания программ с использованием подпрограмм и отладки кода.