РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 9

дисциплина: Архитектура компьютера

Понятие подпрограммы. Отладчик GDB.

Студент: ТУЙИШИМЕ Тьерри

Группа: НКАбд-05-25

Оглавление

1. Цель работы

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

2. Теоретическая часть

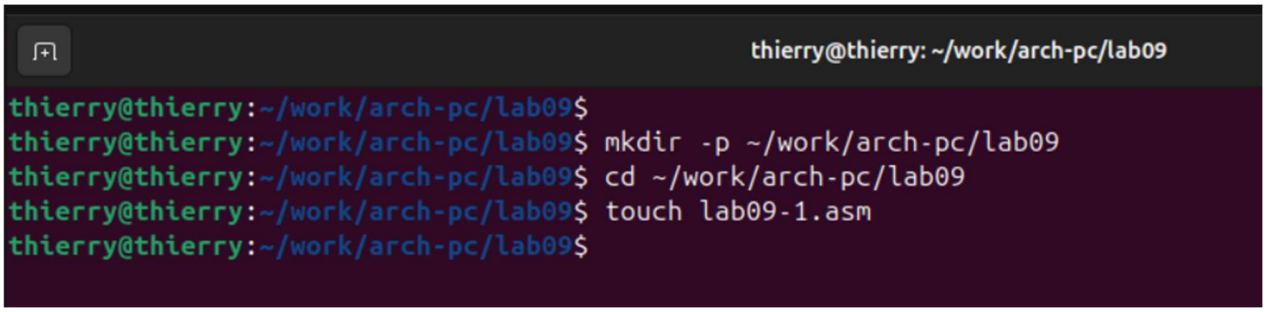
Подпрограмма — это функционально законченный участок кода, который можно многократно вызывать из разных мест программы. Для вызова используется инструкция `call`, для возврата — `ret`.

Отладка — процесс поиска и исправления ошибок в программе. GDB (GNU Debugger) — отладчик, позволяющий управлять выполнением программы, устанавливать точки останова, просматривать и изменять данные.

3. Ход работы

3.1. Реализация подпрограмм в NASM

3.1.1. Создание каталога и файла



3.1.2. Написание программы с подпрограммой `_calcul`

Файл: `lab09-1.asm`

3.1.3. Компиляция и запуск

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

3.2. Отладка программы с помощью GDB

3.2.1. Создание файла `lab09-2.asm`

Файл: `lab09-2.asm`

Open

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

```
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
```

3.2.2. Компиляция с отладочной информацией

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

3.2.3. Запуск GDB

```
thierry@thierry:~/work/arch-pc/lab09$ gdb lab09-2
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
Copyright (C) 2024 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 lab09-2...
(gdb)
```

3.2.4. Установка точки останова и запуск программы

```
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab09-2.asm, line 12.
(gdb) run
Starting program: /home/thierry/work/arch-pc/lab09/lab09-2

This GDB supports auto-downloading debuginfo from the following URLs:
  <https://debuginfod.ubuntu.com>
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

Breakpoint 1, _start () at lab09-2.asm:12
12          mov eax, 4
(gdb)
```

3.2.5. Дизассемблирование

```
(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) 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)
```

3.2.6. Установка второй точки останова

```
(gdb) break *0x8049031
Breakpoint 2 at 0x8049031: file lab09-2.asm, line 25.
(gdb) info breakpoints
Num     Type           Disp Enb Address    What
1       breakpoint     keep y   0x08049000 lab09-2.asm:12
        breakpoint already hit 1 time
2       breakpoint     keep y   0x08049031 lab09-2.asm:25
(gdb)
```

3.2.7. Пошаговое выполнение и просмотр регистров

```
(gdb) stepi
13          mov ebx, 1
(gdb) info registers
eax            0x4                 4
ecx            0x0                 0
edx            0x0                 0
ebx            0x0                 0
esp            0xffffd000          0xffffd000
ebp            0x0                 0x0
esi            0x0                 0
edi            0x0                 0
eip            0x8049005           0x8049005 <_start+5>
eflags         0x10202             [ IF RF ]
cs             0x23                35
ss             0x2b                43
ds             0x2b                43
es             0x2b                43
fs             0x0                 0
gs             0x0                 0
(gdb)
```

3.2.8. Просмотр памяти

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

3.2.9. Изменение данных в памяти

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

3.2.10. Изменение регистров

```
(gdb) set $ebx=2
(gdb) print $ebx
$2 = 2
(gdb)
```

3.3. Обработка аргументов командной строки в GDB

```
thierry@thierry:~/work/arch-pc/lab09$ cp ~/work/arch-pc/lab08/lab8-2.asm ~/work/arch-pc/lab09/lab09-3.asm
thierry@thierry:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-3.lst lab09-3.asm
thierry@thierry:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-3 lab09-3.o
thierry@thierry:~/work/arch-pc/lab09$
```

3.3.2. Запуск GDB с аргументами

```
thierry@thierry:~/work/arch-pc/lab09$ gdb --args lab09-3 аргумент1 аргумент 2 'аргумент 3'
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
Copyright (C) 2024 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)
```

3.3.3. Исследование стека аргументов

```
(gdb) break _start
Breakpoint 1 at 0x8049108: file lab09-3.asm, line 7.
(gdb) run
Starting program: /home/thierry/work/arch-pc/lab09/lab09-3 аргумент1 аргумент 2 аргумент\ 3

This GDB supports auto-downloading debuginfo from the following URLs:
  <https://debuginfod.ubuntu.com>
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

Breakpoint 1, _start () at lab09-3.asm:7
7       pop ecx ; Извлекаем из стека в `ecx` количество аргументов
(gdb) x/x $esp
0xffffcfc0:     0x00000005
(gdb) x/s *(void**)($esp + 4)
0xffffd197:     "/home/thierry/work/arch-pc/lab09/lab09-3"
(gdb) x/s *(void**)($esp + 8)
0xffffd1c0:     "аргумент1"
(gdb)
```

Вывод: Шаг изменения адреса равен 4, потому что в 32-битной архитектуре размер указателя 4 байта.

4. Самостоятельная работа

4.1. Задание 1

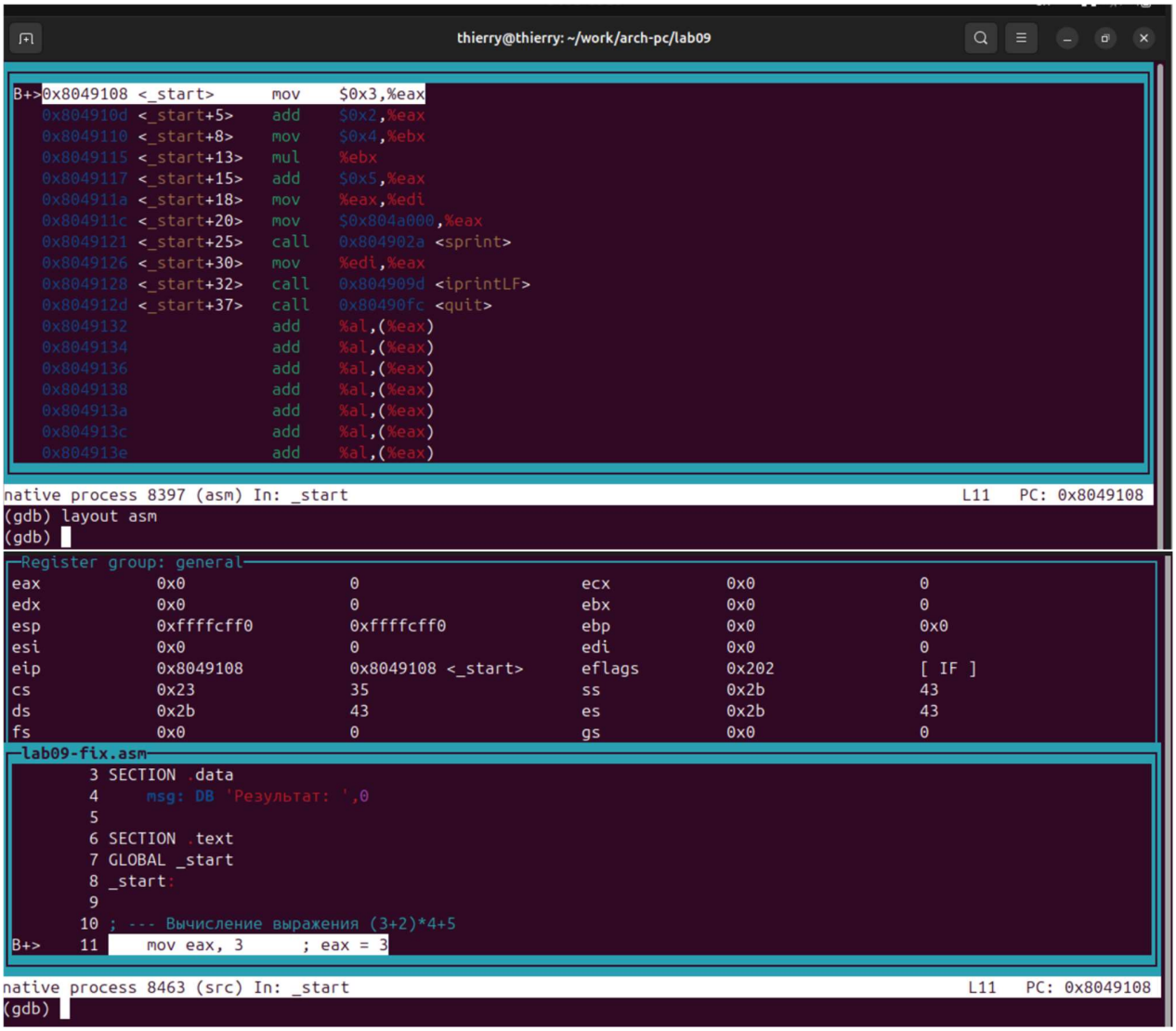
Преобразование программы из лабораторной работы №8 с использованием подпрограммы для вычисления функции.

(Программа представлена в приложении)

4.2. Задание 2

Отладка программы вычисления `(3+2)*4+5`.

Файл: `lab09-fix.asm`



Исправление:Ошибка была в некорректном использовании регистров. После `mul ecx` результат хранится в `eax`, а не в `ebx`.

5. Выводы

В ходе работы были изучены:

- Принципы написания подпрограмм на ассемблере NASM.
- Основы работы с отладчиком GDB.
- Установка точек останова, пошаговое выполнение, просмотр и изменение регистров и памяти.
- Анализ стека аргументов командной строки.

Навыки, полученные в лабораторной работе, позволяют эффективно отлаживать и оптимизировать ассемблерные программы.

6. Приложения

Исходные файлы:

- `lab09-1.asm`
- `lab09-2.asm`
- `lab09-3.asm`
- `lab09-fix.asm`
- Скриншоты выполнения
- Листинги программ