

Отчёт по лабораторной работе №7

дисциплина: Архитектура компьютера

Сычев Егор Олегович

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Ответы на вопросы	13
4	Самостоятельная работа	15
5	Вывод	17

Список иллюстраций

2.1	Команды: mkdir, cd, touch	6
2.2	mcedit	6
2.3	Транслирование, компоновка, запуск	7
2.4	mcedit	7
2.5	Транслирование, компоновка, запуск	7
2.6	Команды: touch, mcedit	7
2.7	mcedit	8
2.8	Транслирование, компоновка, запуск	8
2.9	mcedit	8
2.10	Транслирование, компоновка, запуск	8
2.11	mcedit	9
2.12	Транслирование, компоновка, запуск	9
2.13	Команды: touch, mcedit	9
2.14	mcedit	10
2.15	Транслирование, компоновка, запуск	10
2.16	mcedit	11
2.17	Транслирование, компоновка, запуск	11
2.18	Команды: touch, mcedit	11
2.19	mcedit	12
2.20	Транслирование, компоновка, запуск	12
4.1	Команды: touch, mcedit	15
4.2	mcedit	16
4.3	Транслирование, компоновка, запуск	16

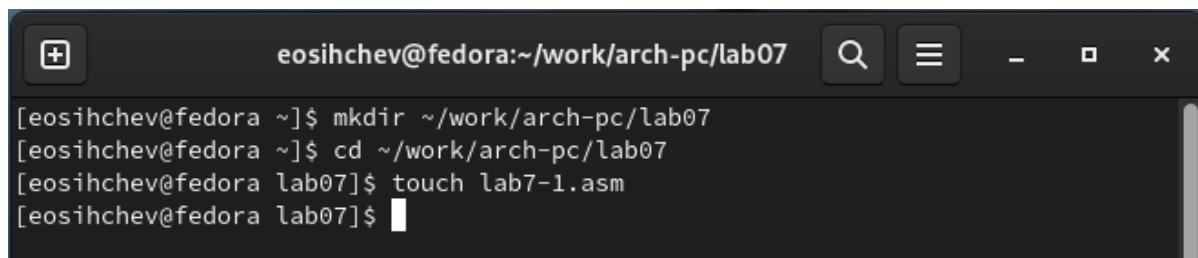
Список таблиц

1 Цель работы

Освоить арифметические инструкции языка ассемблера NASM.

2 Выполнение лабораторной работы

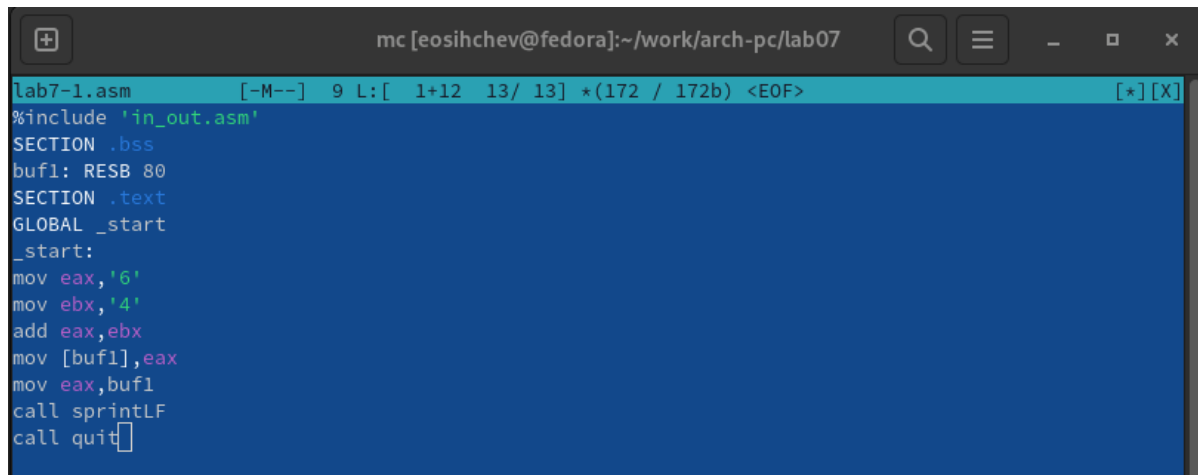
1. Создаем каталог для программ лабораторной работы №7, переходим в него и создаем файл lab7-1.asm.



```
eosihchev@fedora:~/work/arch-pc/lab07
[eosihchev@fedora ~]$ mkdir ~/work/arch-pc/lab07
[eosihchev@fedora ~]$ cd ~/work/arch-pc/lab07
[eosihchev@fedora lab07]$ touch lab7-1.asm
[eosihchev@fedora lab07]$
```

Рис. 2.1: Команды: mkdir, cd, touch

2. Вводим в файл lab7-1.asm текст программы из листинга 7.1.



```
lab7-1.asm  [-M--]  9 L: [ 1+12 13/ 13] *(172 / 172b) <EOF>  [*] [X]
%include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF
call quit
```

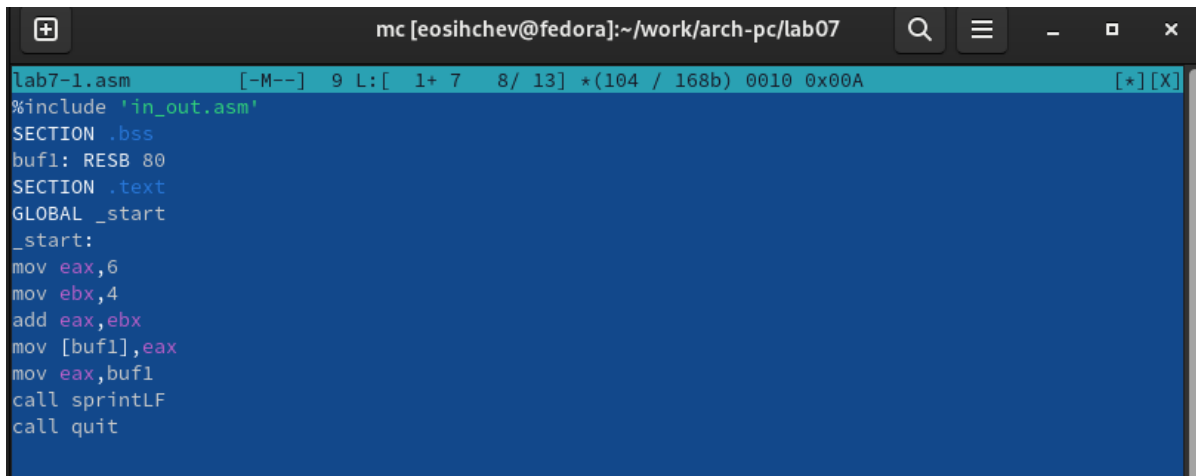
Рис. 2.2: mcedit

3. Создаем исполняемый файл и запускаем его.

```
[eosihchev@fedora lab07]$ nasm -f elf lab7-1.asm
[eosihchev@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[eosihchev@fedora lab07]$ ./lab7-1
j
```

Рис. 2.3: Транслирование, компоновка, запуск

4. Изменяем текст программы и вместо символов, запишем в регистры числа. Создаем исполняемый файл и запускаем его. В данном случае выводится символ с кодом 10 - перенос строки.



```
lab7-1.asm [-M--] 9 L: [ 1+ 7 8/ 13] *(104 / 168b) 0010 0x00A [*] [X]
#include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
mov [buf1],eax
mov eax,buf1
call printf
call _exit
```

Рис. 2.4: mcedit

```
[eosihchev@fedora lab07]$ nasm -f elf lab7-1.asm
[eosihchev@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[eosihchev@fedora lab07]$ ./lab7-1
j

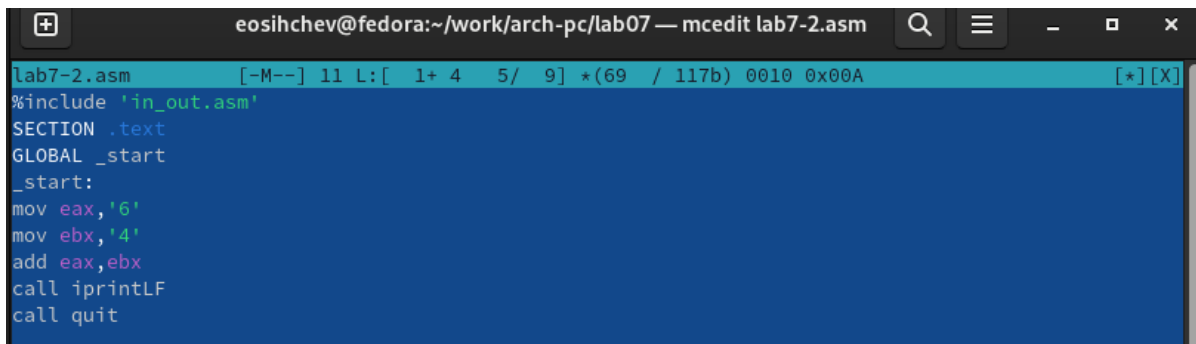
[eosihchev@fedora lab07]$
```

Рис. 2.5: Транслирование, компоновка, запуск

5. Создаем файл lab7-2.asm, вводим в него текст программы из листинга 7.2.

```
[eosihchev@fedora lab07]$ touch lab7-2.asm
[eosihchev@fedora lab07]$ mcedit lab7-2.asm
```

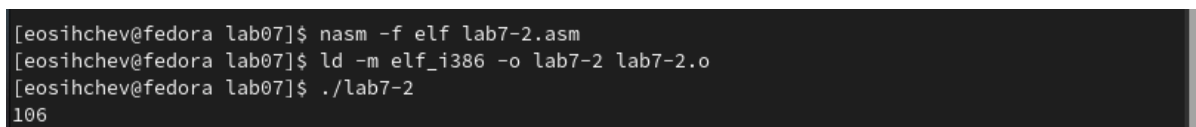
Рис. 2.6: Команды: touch, mcedit



```
lab7-2.asm [-M--] 11 L: [ 1+ 4 5/ 9] *(69 / 117b) 0010 0x00A [*] [X]
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit
```

Рис. 2.7: mcedit

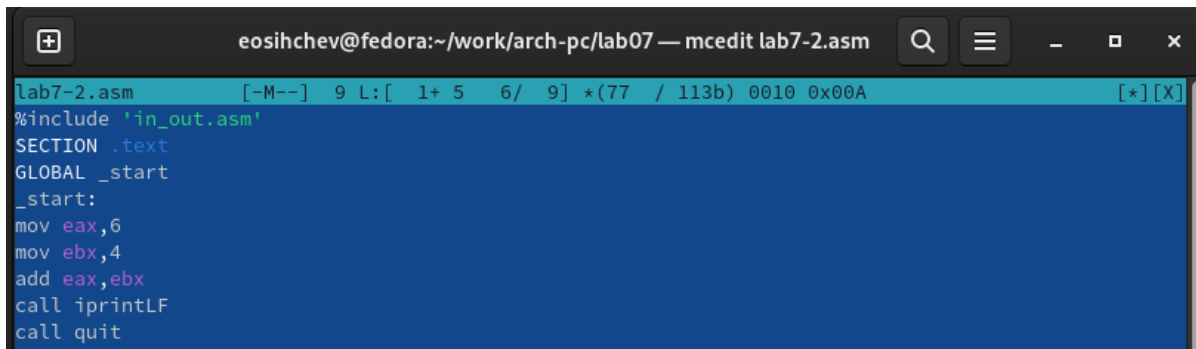
6. Создаем исполняемый файл и запускаем его.



```
[eosihchev@fedora lab07]$ nasm -f elf lab7-2.asm
[eosihchev@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[eosihchev@fedora lab07]$ ./lab7-2
106
```

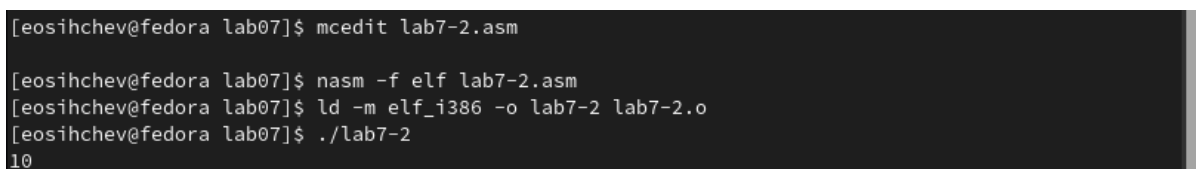
Рис. 2.8: Транслирование, компоновка, запуск

7. Изменяем текст программы аналогично с lab7-1.asm. Создаем исполняемый файл и запускаем его.



```
lab7-2.asm [-M--] 9 L: [ 1+ 5 6/ 9] *(77 / 113b) 0010 0x00A [*] [X]
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

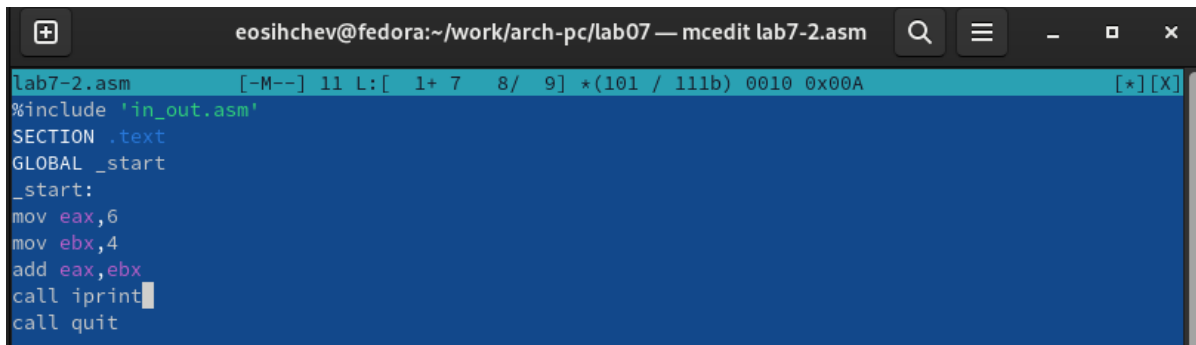
Рис. 2.9: mcedit



```
[eosihchev@fedora lab07]$ mcedit lab7-2.asm
[eosihchev@fedora lab07]$ nasm -f elf lab7-2.asm
[eosihchev@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[eosihchev@fedora lab07]$ ./lab7-2
10
```

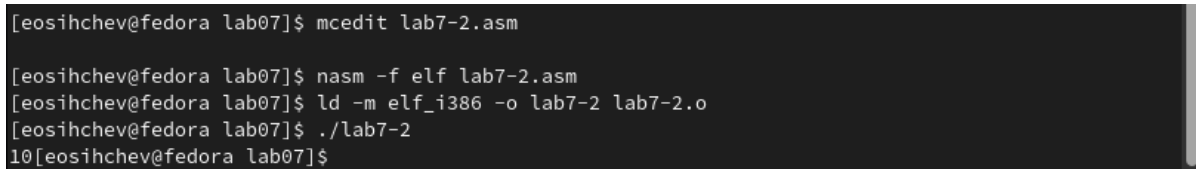
Рис. 2.10: Транслирование, компоновка, запуск

8. Также заменяем функцию `iprintLF` на `iprint`. Создаем исполняемый файл и запускаем его.



```
lab7-2.asm [-M--] 11 L:[ 1+ 7 8/ 9] *(101 / 111b) 0010 0x00A [*] [X]
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit
```

Рис. 2.11: mcedit



```
[eosihchev@fedora lab07]$ mcedit lab7-2.asm
[eosihchev@fedora lab07]$ nasm -f elf lab7-2.asm
[eosihchev@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[eosihchev@fedora lab07]$ ./lab7-2
10[eosihchev@fedora lab07]$
```

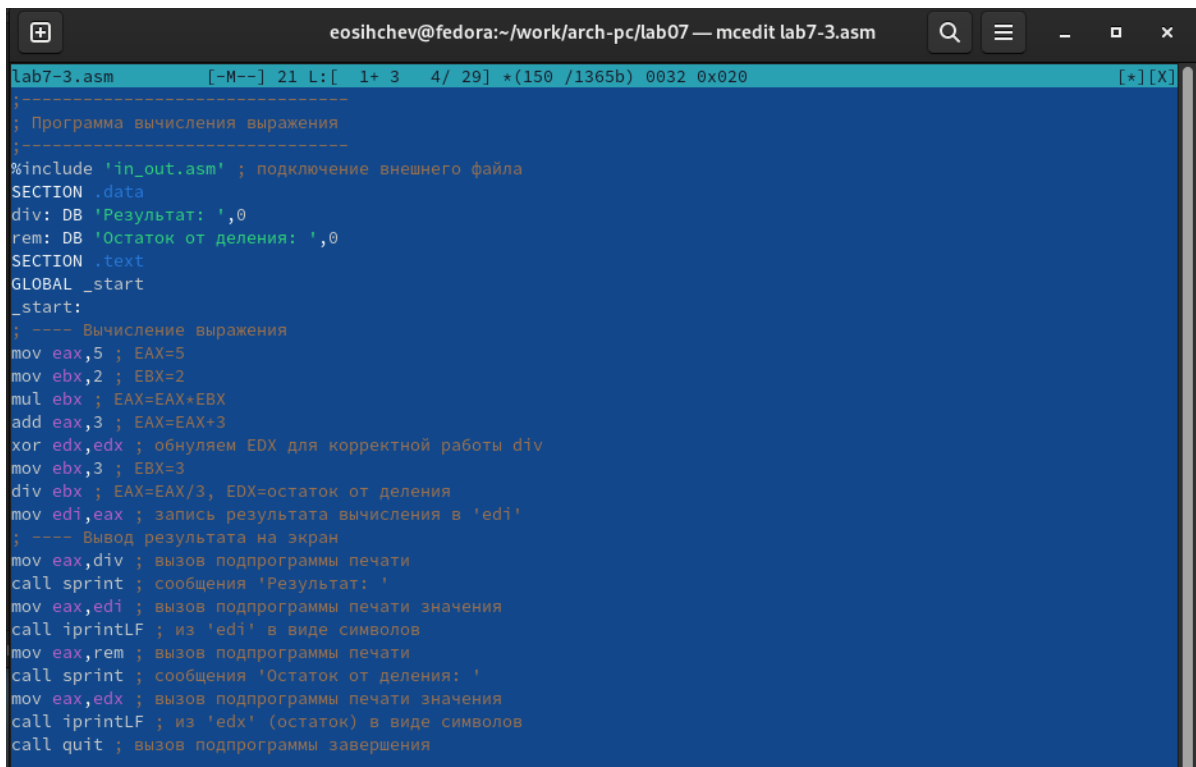
Рис. 2.12: Транслирование, компоновка, запуск

9. Создаем файл `lab7-3.asm`, вводим в него текст программы из листинга 7.3.



```
[eosihchev@fedora lab07]$ touch lab7-3.asm
[eosihchev@fedora lab07]$ mcedit lab7-3.asm
```

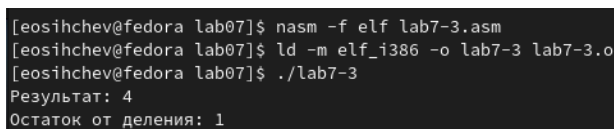
Рис. 2.13: Команды: touch, mcedit



```
lab7-3.asm [-M--] 21 L: [ 1+ 3 4/ 29] *(150 /1365b) 0032 0x020 [*] [X]
;-----
; Программа вычисления выражения
;-----
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,5 ; EAX=5
mov ebx,2 ; EBX=2
mul ebx ; EAX=EAX*EBX
add eax,3 ; EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,3 ; EBX=3
div ebx ; EAX=EAX/3, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 2.14: mcedit

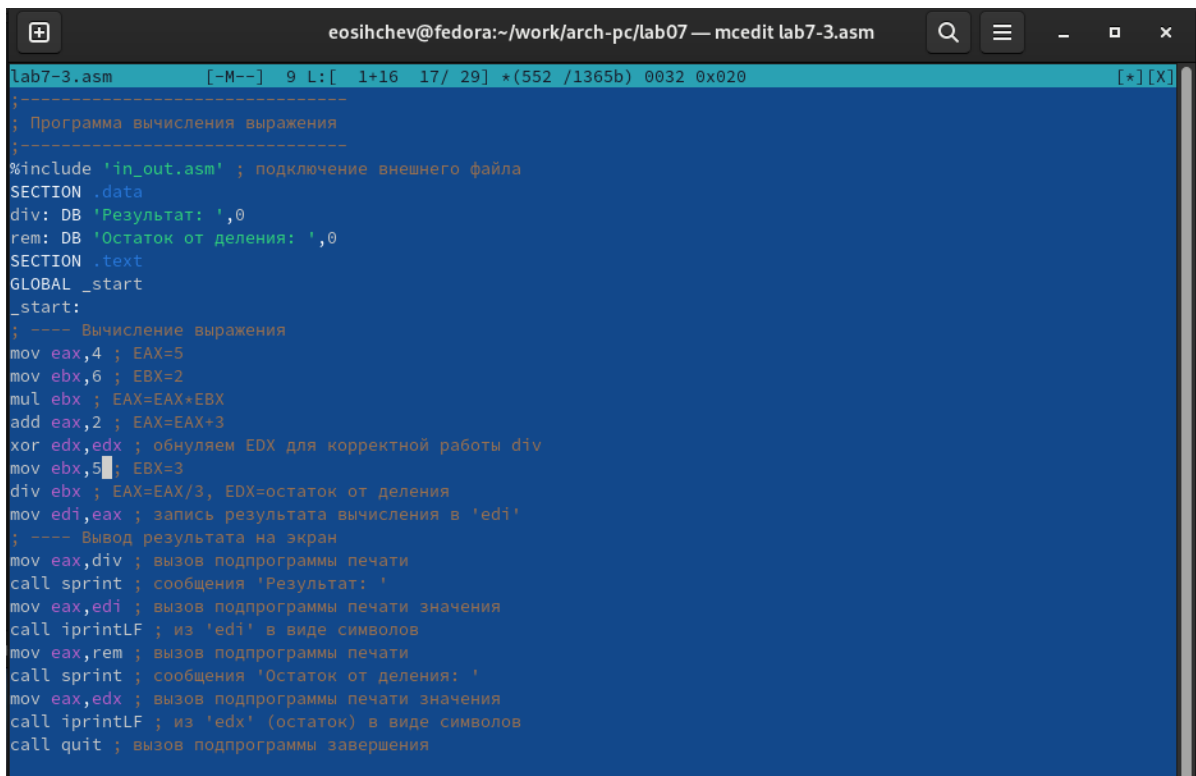
10. Создаем исполняемый файл и запускаем его.



```
[eosihchev@fedora lab07]$ nasm -f elf lab7-3.asm
[eosihchev@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[eosihchev@fedora lab07]$ ./lab7-3
Результат: 4
Остаток от деления: 1
```

Рис. 2.15: Транслирование, компоновка, запуск

11. Изменяем текст программы для вычисления выражения $f(x)=(4*6+2)/5$. Создаем исполняемый файл и запускаем его.



```
lab7-3.asm [-M--] 9 L: [ 1+16 17/ 29] *(552 /1365b) 0032 0x020 [*] [X]
;-----
; Программа вычисления выражения
;-----
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,4 ; EAX=5
mov ebx,6 ; EBX=2
mul ebx ; EAX=EAX*EBX
add eax,2 ; EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,5 ; EBX=3
div ebx ; EAX=EAX/3, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 2.16: mcedit



```
[eosihchev@fedora lab07]$ mcedit lab7-3.asm
[eosihchev@fedora lab07]$ nasm -f elf lab7-3.asm
[eosihchev@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[eosihchev@fedora lab07]$ ./lab7-3
Результат: 5
Остаток от деления: 1
```

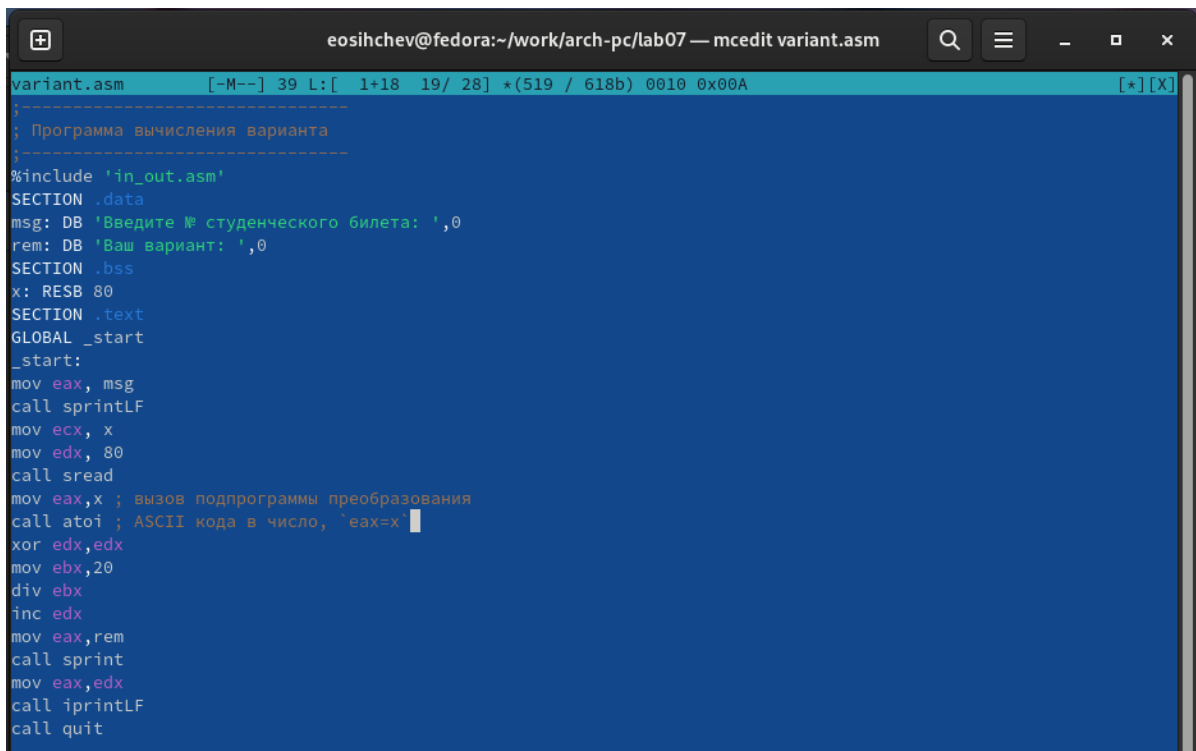
Рис. 2.17: Транслирование, компоновка, запуск

12. Создаем файл variant.asm, вводим в него текст программы из листинга 7.4.



```
[eosihchev@fedora lab07]$ touch variant.asm
[eosihchev@fedora lab07]$ mcedit variant.asm
```

Рис. 2.18: Команды: touch, mcedit

A screenshot of a text editor window titled 'eosihchev@fedora:~/work/arch-pc/lab07 — mcedit variant.asm'. The editor shows assembly code for a program that calculates a variant. The code includes comments in Russian, such as 'Программа вычисления варианта'. It defines data sections for messages and a register section for variables like 'x' and 'rem'. The main logic involves reading input, converting it to a number, and performing calculations to determine the final variant number.

```
variant.asm [-M--] 39 L:[ 1+18 19/ 28] *(519 / 618b) 0010 0x00A [*][X]
; Программа вычисления варианта
-----
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите № студенческого билета: ',0
rem: DB 'Ваш вариант: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintf
mov ecx, x
mov edx, 80
call sread
mov eax, x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, 'eax=x'
xor edx, edx
mov ebx, 20
div ebx
inc edx
mov eax, rem
call sprintf
mov eax, edx
call iprintLF
call quit
```

Рис. 2.19: mcedit

13. Создаем исполняемый файл и запускаем его.

A screenshot of a terminal window showing the steps to compile and run the assembly program. The user runs 'nasm -f elf variant.asm' to create an object file, then 'ld -m elf_i386 -o variant variant.o' to create an executable. Finally, they run './variant', which prompts for a student ID and displays the calculated variant number.

```
[eosihchev@fedora lab07]$ nasm -f elf variant.asm
[eosihchev@fedora lab07]$ ld -m elf_i386 -o variant variant.o
[eosihchev@fedora lab07]$ ./variant
Введите № студенческого билета:
1132226469
Ваш вариант: 10
```

Рис. 2.20: Транслирование, компоновка, запуск

3 Ответы на вопросы

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант’?
 - `mov eax,rem`
 - `call sprint`
2. Для чего используются следующие инструкции: `mov eax, x` | `mov edx, 80` | `call sread` ?
 - `mov eax, x` : записывает данные из `x` в регистр `eax`
 - `mov edx, 80` : указывает длину переменной `x`
 - `call sread` : считывает введенную информацию
3. Для чего используется инструкция “`call atoi`”?
 - Преобразует ASCII код в число
4. Какие строки листинга 7.4 отвечают за вычисления варианта?
 - `xor edx,edx`
 - `mov ebx,20`
 - `div ebx`
 - `inc edx`
5. В какой регистр записывается остаток от деления при выполнении инструкции “`div ebx`”?

- `edx`

6. Для чего используется инструкция `inc edx`?


- Прибавляет единицу

7. Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений?

- `mov eax,edx`
- `call iprintLF`

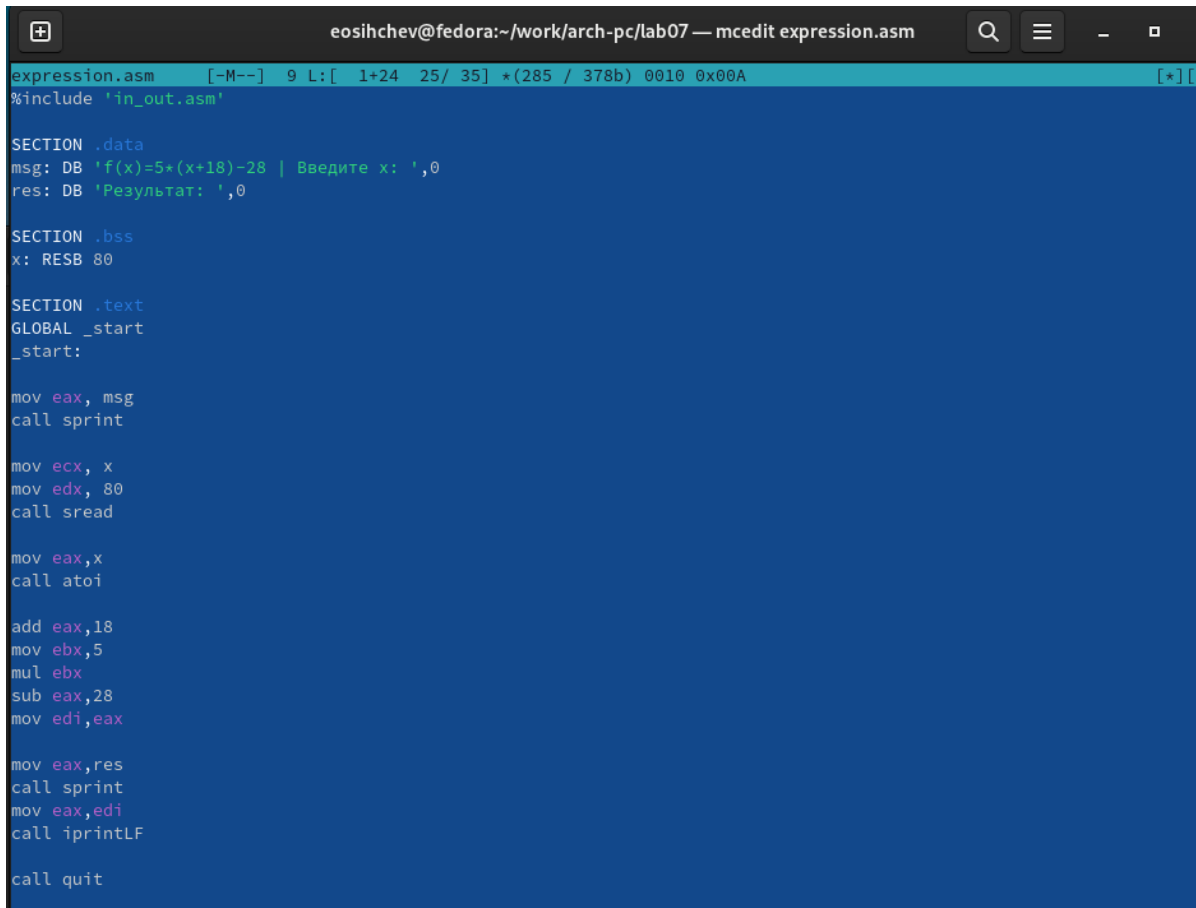
4 Самостоятельная работа

1. Создаем файл `expression.asm`, вводим в него текст программы для вычисления выражения $f(x)=5*(x+18)-28$.

A terminal window with a dark background and light gray text. It shows two lines of commands being executed in a shell. The first line is '[eosihchev@fedora lab07]\$ touch expression.asm' and the second line is '[eosihchev@fedora lab07]\$ mcedit expression.asm'.

```
[eosihchev@fedora lab07]$ touch expression.asm
[eosihchev@fedora lab07]$ mcedit expression.asm
```

Рис. 4.1: Команды: `touch`, `mcedit`



```
expression.asm  [-M--]  9 L: [ 1+24 25/ 35] *(285 / 378b) 0010 0x00A  [*]
#include 'in_out.asm'

SECTION .data
msg: DB 'f(x)=5*(x+18)-28 | Введите x: ',0
res: DB 'Результат: ',0

SECTION .bss
x: 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

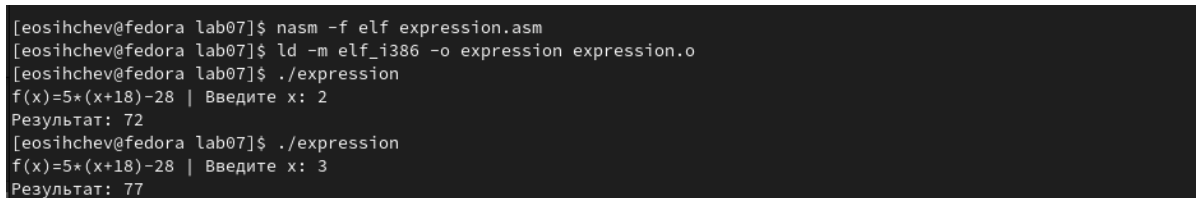
add eax, 18
mov ebx, 5
mul ebx
sub eax, 28
mov edi, eax

mov eax, res
call sprint
mov eax, edi
call iprintLF

call quit
```

Рис. 4.2: mcedit

13. Создаем исполняемый файл и запускаем его. Для проверки возьмем 2 значения: $x_1=2$ | $x_2=3$.



```
[eosihchev@fedora lab07]$ nasm -f elf expression.asm
[eosihchev@fedora lab07]$ ld -m elf_i386 -o expression expression.o
[eosihchev@fedora lab07]$ ./expression
f(x)=5*(x+18)-28 | Введите x: 2
Результат: 72
[eosihchev@fedora lab07]$ ./expression
f(x)=5*(x+18)-28 | Введите x: 3
Результат: 77
```

Рис. 4.3: Транслирование, компоновка, запуск

5 Вывод

Я освоил арифметические инструкции языка ассемблера NASM.