Отчёт по лабораторной работе №6

Дисциплина: Архитектура компьютера

Калашникова Ольга Сергеевна

Содержание

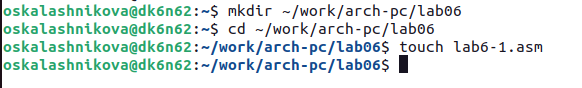
# 1 Цель работы

Освоение арифметческих инструкций языка ассемблера NASM

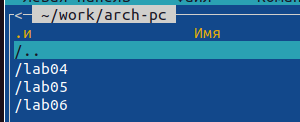
# 2 Выполнение лабораторной работы

## 2.1 Символьные и численные данные в NASM

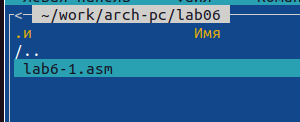
Создаём каталог для программам лабораторной работы № 6 (при помощи команды mkdir ~/work/arch-pc/lab06), переходим в него (при помощи команды cd ~/work/arch-pc/lab06) и создаём файл lab6-1.asm (при помощи команды touch lab6-1.asm)(рис. ??),(рис. ??),(рис. ??)



Создание папки, её открытие и создание файла

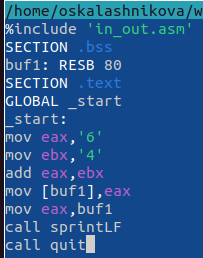


Проверка наличия папки

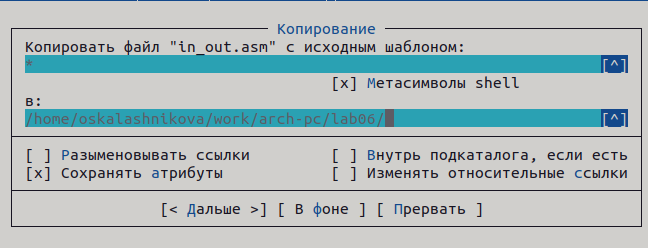


Проверка наличия файла

Введём в файл lab6-1.asm текст программы из листинга 6.1., для корректной работы нужно переместить файл “in\_out.asm” в тот же каталог, где лежит и файл с текстом программы.(рис. ??),(рис. ??)

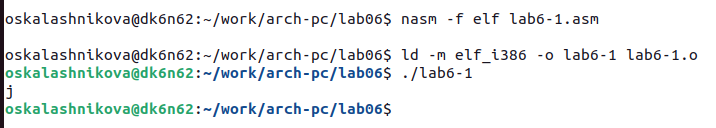


Текст программы



Перемещение файла “in\_out.asm”

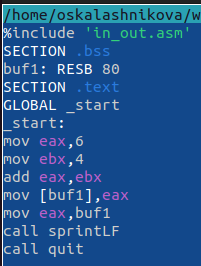
Создаем исполняемый файл и запускаем его (компиляция: nasm -f elf lab6-1.asm , ld -m elf\_i386 -o lab6-1 lab6-1.o , запуск: ./lab6-1) (рис. ??)



Создание и запуск исполняемого файла

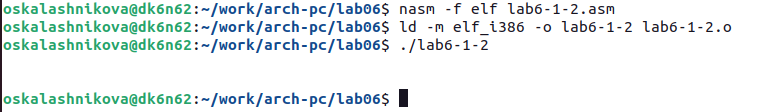
Программа выводит символ j, потому что этот символ соответствует по системе ASCII сумме двоичных кодов символов 4 и 6 (52+54=106).

Меняем текст программы (вместо символов, записываем в регистры числа) (рис. ??)



Изменённый текст программы

Создаём исполняемый файл и запускаем его (компиляция: nasm -f elf lab6-1-2.asm , ld -m elf\_i386 -o lab6-1-2 lab6-1-2.o , запуск: ./lab6-1-2) (рис. ??)



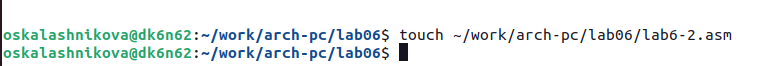
Создание и запуск исполняемого файла

Программа выполняет перевод строки, потому что символ перевода строки соответствует по системе ASCII сумме двоичных кодов чисел 4 и 6 (4+6=10).(рис. ??)

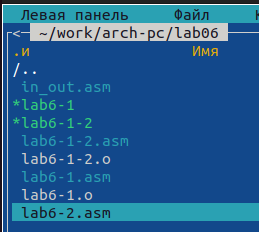
Символ 10 в системе ASCII

Символ 10 в системе ASCII

Создаём файл “lab6-2.asm” в каталоге ~/work/arch-pc/lab06 (touch ~/work/arch-pc/lab06/lab6-2.asm) (рис. ??), (рис. ??)

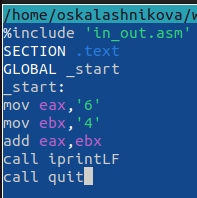


Создание файла



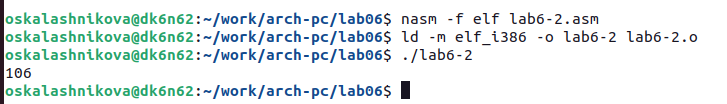
Проверка наличия файла

Вводим в файл текст другой программы для вывода значения регистра eax, которая использует iprintLF (рис. ??)



Текст программы

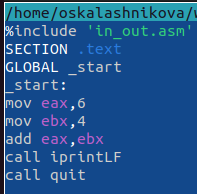
Создаём исполняемый файл и запускаем его (компиляция: nasm -f elf lab6-2.asm , ld -m elf\_i386 -o lab6-2 lab6-2.o , запуск: ./lab6-2) (рис. ??)



Создание и запуск исполняемого файла

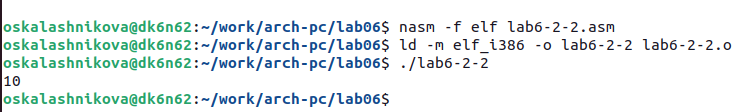
Получаем число 106. Команда add складывает коды символов ‘6’ и ‘4’ (54+52=106). В отличии от программы из листинга 6.1, функция iprintLF преобразует код в число.

Заменяем в тексте программы в файле lab6-2.asm символы “6” и “4” на числа 6 и 4 (рис. ??)



Изменённый текст программы

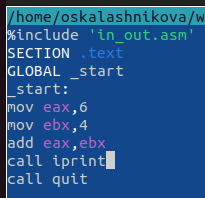
Создаём исполняемый файл и запускаем его (компиляция: nasm -f elf lab6-2-2.asm , ld -m elf\_i386 -o lab6-2-2 lab6-2-2.o , запуск: ./lab6-2-2) (рис. ??)



Создание и запуск исполняемого файла

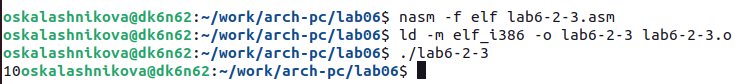
Получаем число 10. Складываем 6 и 4 (6+4=10) , с помощью функции iprintLF, преобразуем код в число и выводим его на экран.

Заменяем в тексте программы в файле lab6-2-2.asm iprintLF на iprint (рис. ??)



Изменённый текст программы

Создаём исполняемый файл и запускаем его (компиляция: nasm -f elf lab6-2-3.asm , ld -m elf\_i386 -o lab6-2-3 lab6-2-3.o , запуск: ./lab6-2-3) (рис. ??)



Создание и запуск исполняемого файла

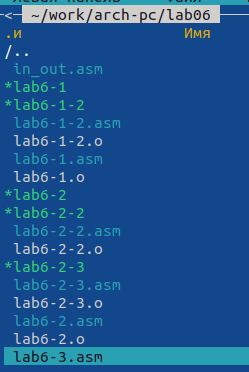
iprint не выполняет перевод строки, а только выводит число на экран, а iprintLF вызывает iprint и добавляет символ перевода строки

## 2.2 Выполнение арифметических операций в NASM

Создаём файл “lab6-3.asm” в каталоге ~/work/arch-pc/lab06 (touch ~/work/arch-pc/lab06/lab6-3.asm) (рис. ??), (рис. ??)

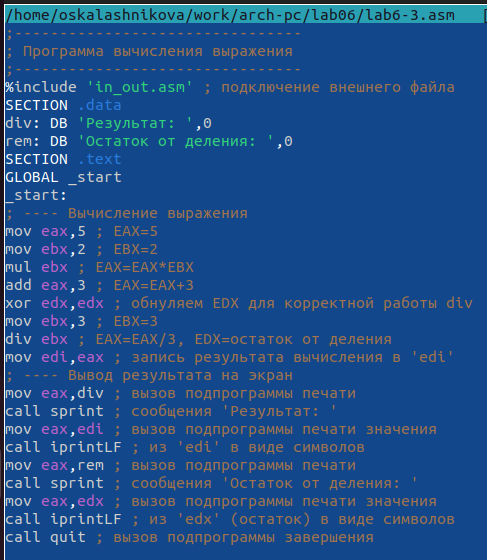
Создание файла

Создание файла



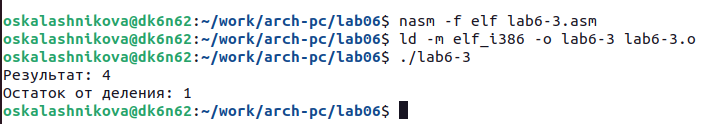
Проверка наличия файла

Вводим в созданный файл текст программы для вычисления значения выражения f(x) = (5 \* 2 + 3)/3 (рис. ??)



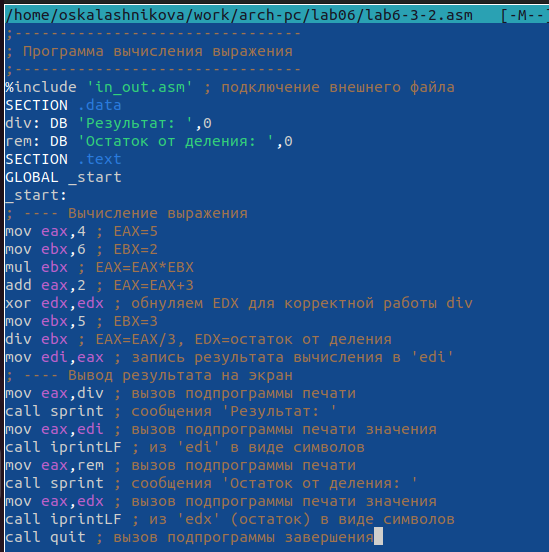
Текст программы

Создаём исполняемый файл и запускаем его (компиляция: nasm -f elf lab6-3.asm , ld -m elf\_i386 -o lab6-3 lab6-3.o , запуск: ./lab6-3) (рис. ??)



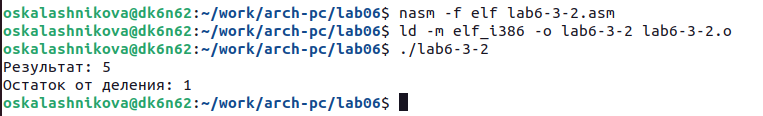
Создание и запуск исполняемого файла

Изменяем программу так, чтобы она вычисляла значение выражения f(x) = (4 \* 6 + 2)/5 (меняются только цифры) (рис. ??)



Изменённый текст программы

Создаём исполняемый файл и запускаем его (компиляция: nasm -f elf lab6-3-2.asm , ld -m elf\_i386 -o lab6-3-2 lab6-3-2.o , запуск: ./lab6-3-2) (рис. ??)



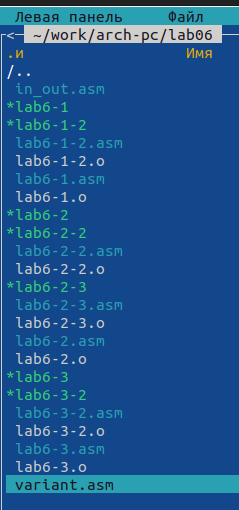
Создание и запуск исполняемого файла

Считаем значения вручную и приходим к выводу, что программа работает верно

Создаём файл “variant.asm” в каталоге ~/work/arch-pc/lab06 (touch ~/work/arch-pc/lab06/variant.asm) (рис. ??), (рис. ??)

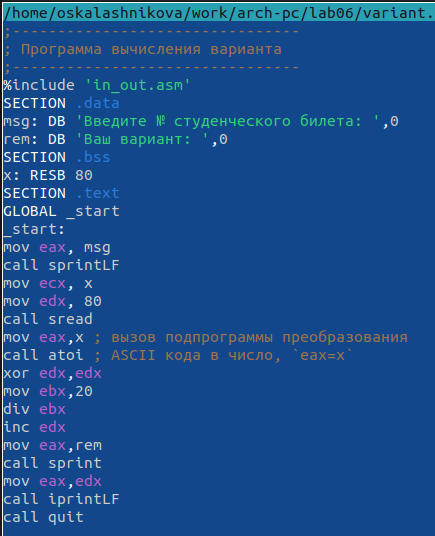
Создание файла

Создание файла



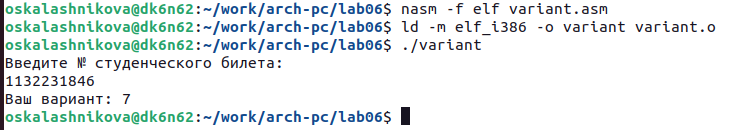
Проверка наличия файла

Вводим в файл текст программы для вычисления варианта задания по номеру студенческого билета (рис. ??)



Текст программы

Создаём исполняемый файл и запускаем его (компиляция: nasm -f elf variant.asm , ld -m elf\_i386 -o variant variant.o , запуск: ./variant) (рис. ??)



Создание и запуск исполняемого файла

При вводе номера своего студенческого я получила 7 вариант

Ответы на вопросы:

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

mov eax,rem  
call sprint

(Так как rem содержит ‘Ваш вариант:’, а call sprint отвечает за вывод на экран сообщения) 2) Для чего используется следующие инструкции?

mov ecx, x  
mov edx, 80  
call sread

Они используются для того, чтобы ввести значение х с клавиатуры (mov ecx, x помещает адрес x в регистр ecx; mov edx, 80 записывает в регистр edx длину строки; call sread вызывает функцию для ввода сообщения с клавиатуры) 3) Для чего используется инструкция “call atoi”? atoi преобразует ascii-код символа в целое число и записывает результат в регистр eax, call atoi вызывает данную функцию 4) Какие строки листинга 6.4 отвечают за вычисления варианта?

xor edx,edx  
mov ebx,20  
div ebx  
inc edx

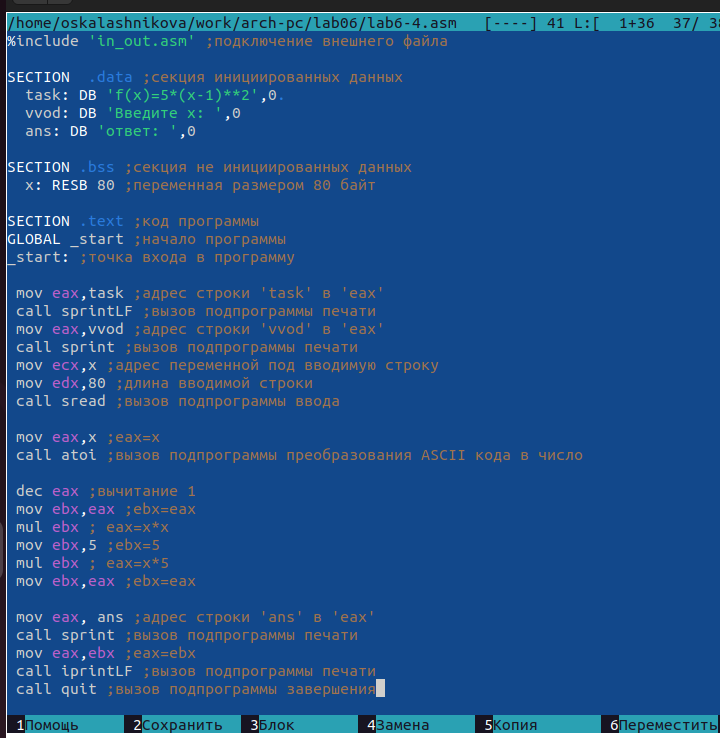
(xor edx,edx обнуляет edx для корректной работы div; mov ebx,20 записыват 20 в регистр ebx; div ebx производит деление eax = eax/20 и записывает остаток в edx; inc edx прибавляет к остатку еденицу, чтобы не было 0 варианта) 5) В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”? В регистр edx 6) Для чего используется инструкция “inc edx”? Для увеличения значения edx на еденицу 7) Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений?

mov eax,edx  
call iprintLF

(mov eax,edx записывает в eax значение вычисленное в edx; call iprintLF вызывает функцию, которая выводит ответ преобразовав его в число)

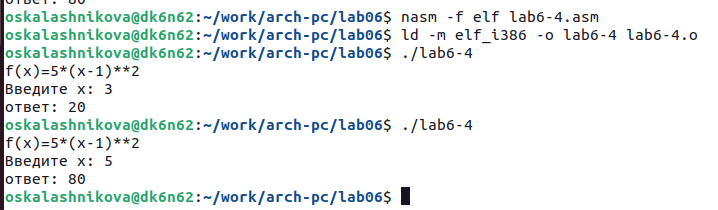
# 3 Задание для самостоятельной работы

Открываем созданный файл lab6-4.asm, вводим в него текст программы для вычисления значения выражения f(x)=5\*(x-1)\*\*2 (рис. ??)



Текст программы

Создаём исполняемый файл и запускаем его (компиляция: nasm -f elf lab6-4.asm , ld -m elf\_i386 -o lab6-4 lab6-4.o , запуск: ./lab6-4) (рис. ??)



Создание и запуск исполняемого файла

Текст программы:

%include 'in\_out.asm' ;подключение внешнего файла  
SECTION .data ;секция инициированных данных  
 task: DB 'f(x)=5\*(x-1)\*\*2',0   
 vvod: DB 'Введите x: ',0  
 ans: DB 'ответ: ',0  
SECTION .bss ;секция не инициированных данных  
 x: RESB 80 ;переменная размером 80 байт  
SECTION .text ;код программы  
GLOBAL \_start ;начало программы  
\_start: ;точка входа в программу  
 mov eax,task ;адрес строки 'task' в 'eax'  
 call sprintLF ;вызов подпрограммы печати  
 mov eax,vvod ;адрес строки 'vvod' в 'eax'  
 call sprint ;вызов подпрограммы печати  
 mov ecx,x ;адрес переменной под вводимую строку  
 mov edx,80 ;длина вводимой строки  
 call sread ;вызов подпрограммы ввода  
 mov eax,x ;eax=x  
 call atoi ;вызов подпрограммы преобразования ASCII кода в число  
 dec eax ;вычитание 1  
 mov ebx,eax ;ebx=eax  
 mul ebx ; eax=x\*х  
 mov ebx,5 ;ebx=5  
 mul ebx ; eax=x\*5  
 mov ebx,eax ;ebx=eax  
 mov eax, ans ;адрес строки 'ans' в 'eax'  
 call sprint ;вызов подпрограммы печати  
 mov eax,ebx ;eax=ebх  
 call iprintLF ;вызов подпрограммы печати  
 call quit ;вызов подпрограммы завершения

# 4 Выводы

При выполнении данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM