Лабораторная работа №9

Воинов Кирилл Викторович

Содержание

# 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

1. Реализация подпрограмм в NASM
2. Создаю каталог для выполнения лабораторной работы No 9, перехожу в него и создаю файл lab09-1.asm.(рис. [1](#fig:001)).

Figure 1: Создание каталога, переход в него и создание файла

Figure 1: Создание каталога, переход в него и создание файла

1. Ввожу в файл lab09-1.asm текст программы из листинга 9.1.(рис. [2](#fig:002)).

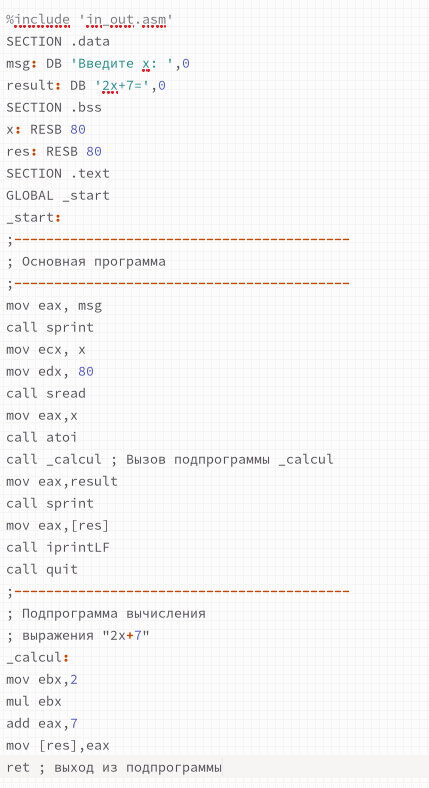


Figure 2: Текст программы

Создаю исполняемый файл и проверьте его работу.(рис. [3](#fig:003)).

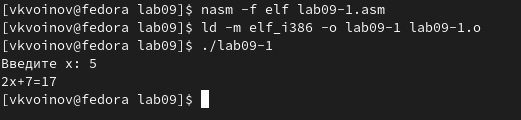


Figure 3: Работа программы

Изменяю текст программы, добавив подпрограмму \_subcalcul в подпрограмму \_calcul, для вычисления выражения f(g(x)), где x вводится с клавиатуры, f(x) = 2x + 7, g(x) = 3x - 1. Т.е. x передается в подпрограмму \_calcul из нее в подпрограмму \_subcalcul, где вычисляется выражение g(x), результат возвращается в \_calcul и вычисляется выражение f(g(x)). Результат возвращается в основную программу для вывода результата на экран.(рис. [4](#fig:004)) и (рис. [4](#fig:004)).

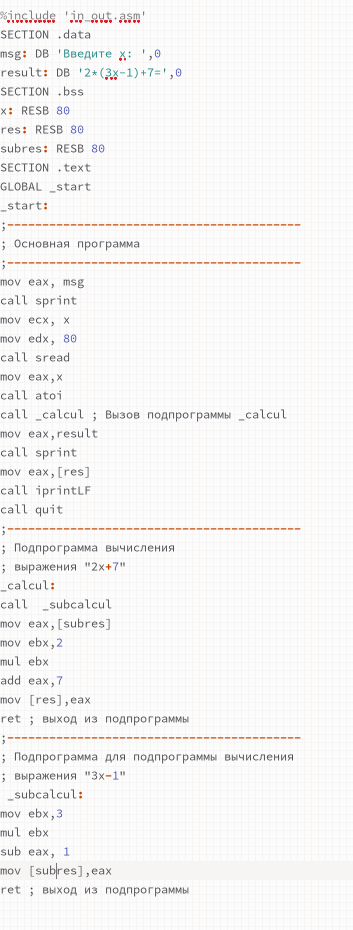


Figure 4: Текст измененной программы

[Работа измененной программы]](image/5.png){#fig:005 width=70%}

1. Отладка программам с помощью GDB
2. Создаю файл lab09-2.asm с текстом программы из Листинга 9.2. (Программа печати сообщения Hello world!).(рис. [5](#fig:006)) и (рис. [6](#fig:007)).

Figure 5: Создание файла

Figure 5: Создание файла

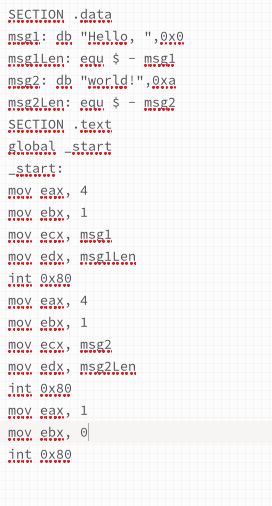


Figure 6: Текст программы

Получаю исполняемый файл и загружаю его в отладчик gdb.(рис. [7](#fig:008)).

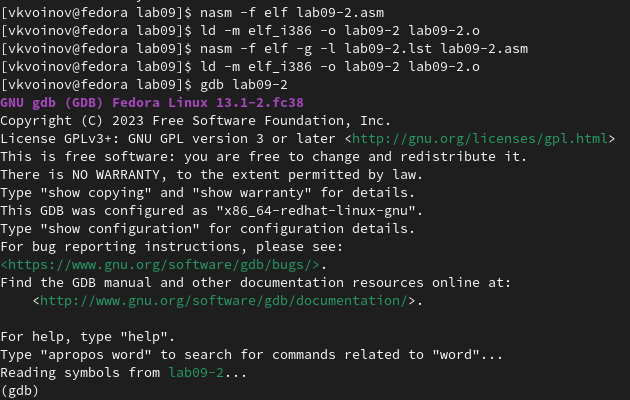


Figure 7: Получение исполняемого файла и загрузка его в отладчик gdb.

Проверяю работу программы, запустив ее в оболочке GDB с помощью команды run.(рис. [8](#fig:009)).

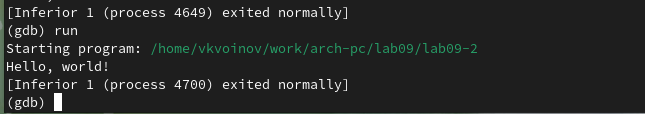


Figure 8: Проверка работы программы

Устанавливаю брейкпоинт на метку \_start и запускаю программу.(рис. [9](#fig:010)).

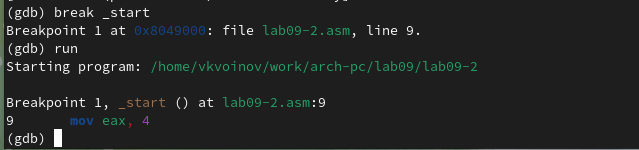


Figure 9: Установка брейкпоинта и запуск программы

С помощью команды disassemble начиная с метки \_start смотрю дисассимилированный код программы.(рис. [10](#fig:011)).



Figure 10: Дисассимилированный код программы

Переключаюсь на отображение команд с Intel’овским синтаксисом.(рис. [11](#fig:012)).



Figure 11: Отображение команд с Intel’овским синтаксисом

Различия отображения синтаксиса машинных команд в режимах ATT и Intel: Различия присутствуют лишь конце строчек. Перед операндами и адресами в первом случае ставятся % и $ соответственно и первым стоит адрес, а во втором таких символов нет и первым стоит адрес.(например $0x4,%eax и eax,0x4)

Включаю режим псевдографики для более удобного анализа программы.(рис. [12](#fig:013)).

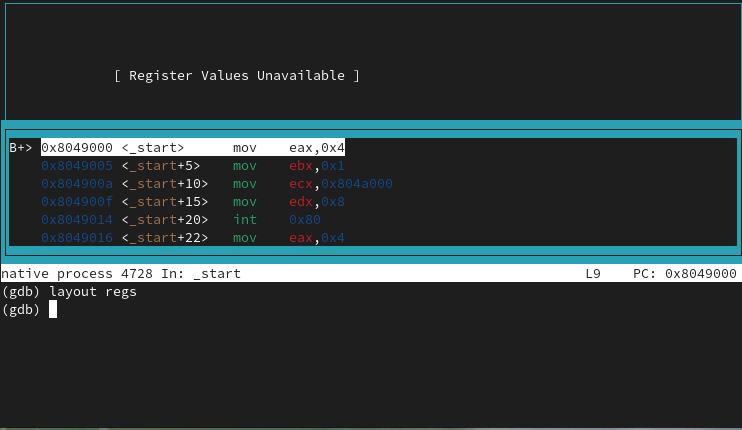


Figure 12: Режим псевдографики

2)Проверяю наличие брейкпоинта по имени метки (\_start) и устанавливаю новую по адресу предпоследней инструкции.(рис. [13](#fig:014)).

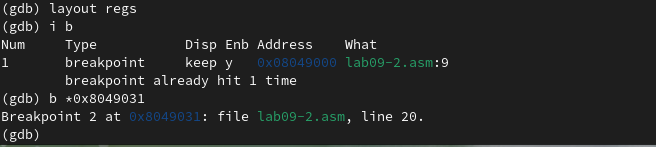


Figure 13: Проверяю наличие брейкпоинта и установка нового

Просматриваю информацию о всех установленных брейкпоинтах.(рис. [14](#fig:015)).

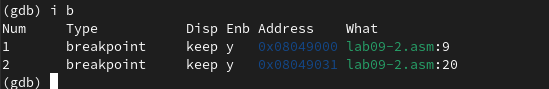


Figure 14: Информация о брейкпоинтах

При выполнении 5 инструкций изменяются регистры eax, ebx, ecx, edx, eax.

Смотрю значение переменной msg1.(рис. [15](#fig:016)).

Figure 15: Значение переменной msg1

Figure 15: Значение переменной msg1

Изменяю первый символ переменной msg1.(рис. [16](#fig:017)).

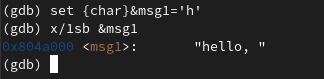


Figure 16: Изменение первого символа переменной msg1

Изменяю первый символ переменной msg2.(рис. [17](#fig:018)).

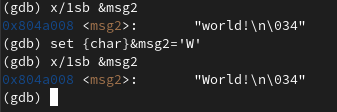


Figure 17: Изменение первого символа переменной msg2

С помощью команды set изменяю значение регистра ebx.(рис. [18](#fig:019)).

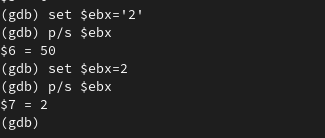


Figure 18: Изменение значения регистра ebx

Разница возникает из за того, что в первом случае 2 читается как символ а во втором как число.

Завершаю выполнение программы и выхожу из GDB.

1. Копирую файл lab8-2.asm, созданный при выполнении лабораторной работы No8, с программой выводящей на экран аргументы командной строки (Листинг 8.2) в файл с именем lab09-3.asm. (рис. [19](#fig:021)).

Figure 19: Копия файла lab8-2.asm

Figure 19: Копия файла lab8-2.asm

Создаю исполняемый файл.(рис. [20](#fig:022)).

Figure 20: Создание исполняемого файла

Figure 20: Создание исполняемого файла

Загружаю исполняемый файл в отладчик, указав аргументы. (рис. [21](#fig:023)).

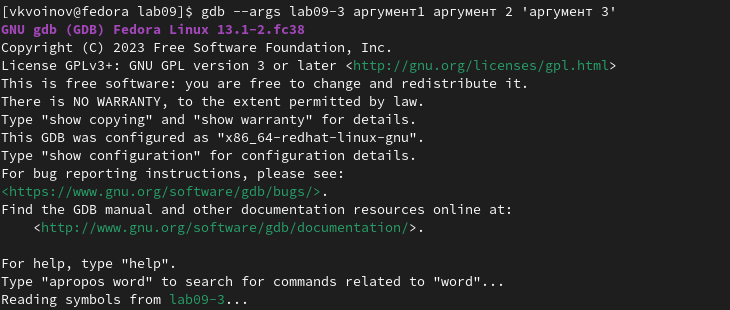


Figure 21: Загрузка исполняемого файла

Устанавливаю точку остановки перед первой инструкцией в программе.(рис. [22](#fig:024)).

Figure 22: Точка остановки

Figure 22: Точка остановки

Запускаю программу и просматриваю позиции стека.(рис. [23](#fig:025)).

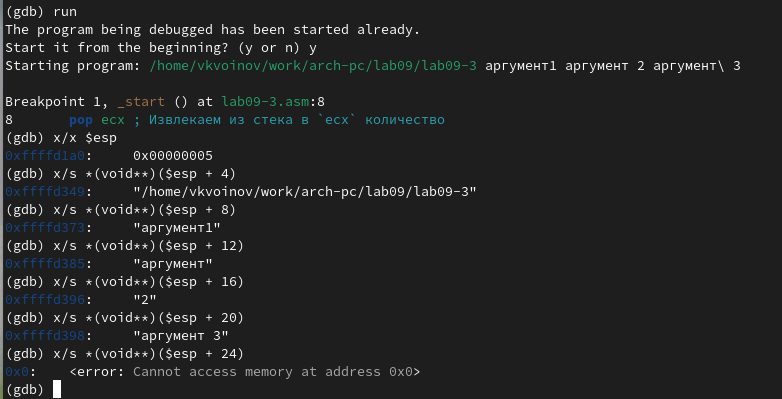


Figure 23: Запуск программы и просмотр элементов стека.

# 3 Задание для самостоятельной работы

1.Преобразовываю программу из лабораторной работы No8 (Задание No1 для самостоятельной работы), реализовав вычисление значения функции f(x) как подпрограмму.(рис. [24](#fig:026)) и (рис. [25](#fig:027)).

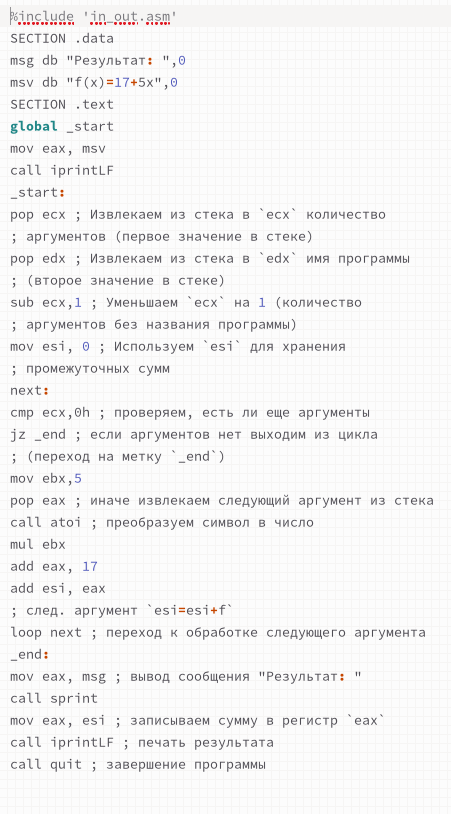


Figure 24: Текст программы

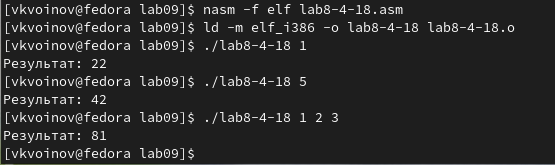


Figure 25: Работа программы

В листинге 9.3 приведена программа вычисления выражения (3+2)\*4+5. При запуске данная программа дает неверный результат. Проверьте это. С помощью отладчика GDB, анализируя изменения значений регистров, определите ошибку и исправьте ее.(рис. [27](#fig:029)).

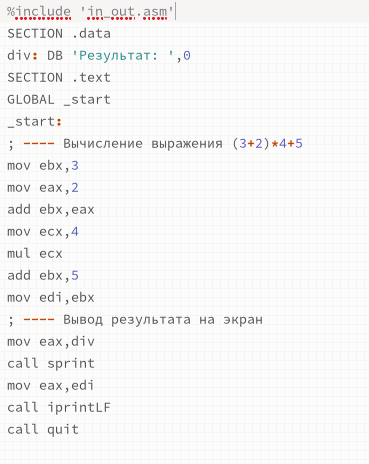


Figure 26: Текст программы

С помощью отладчика я поочередно проверял каждую строку и изменение регистров и заметил, что при умножении на ecx, ebx не изменяется, а изменяется eax. (рис. [27](#fig:029)) и (рис. [28](#fig:030)).

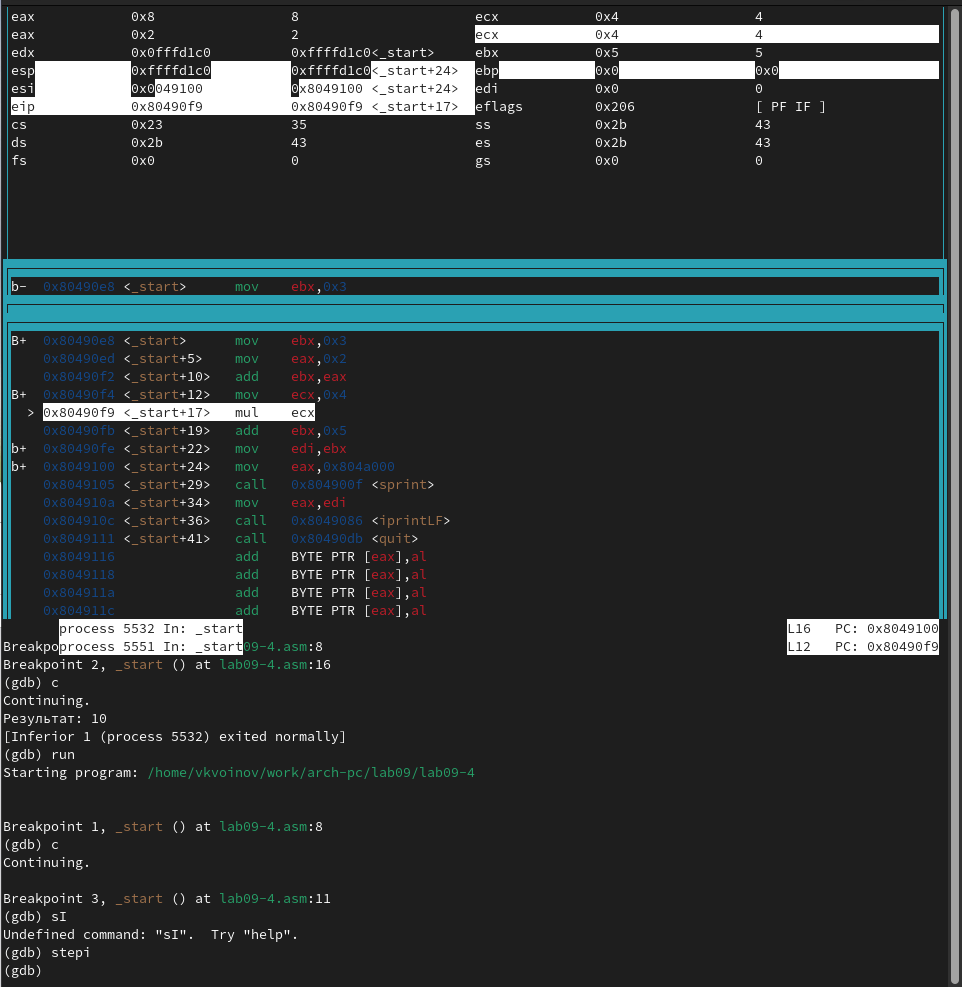


Figure 27: До умножения

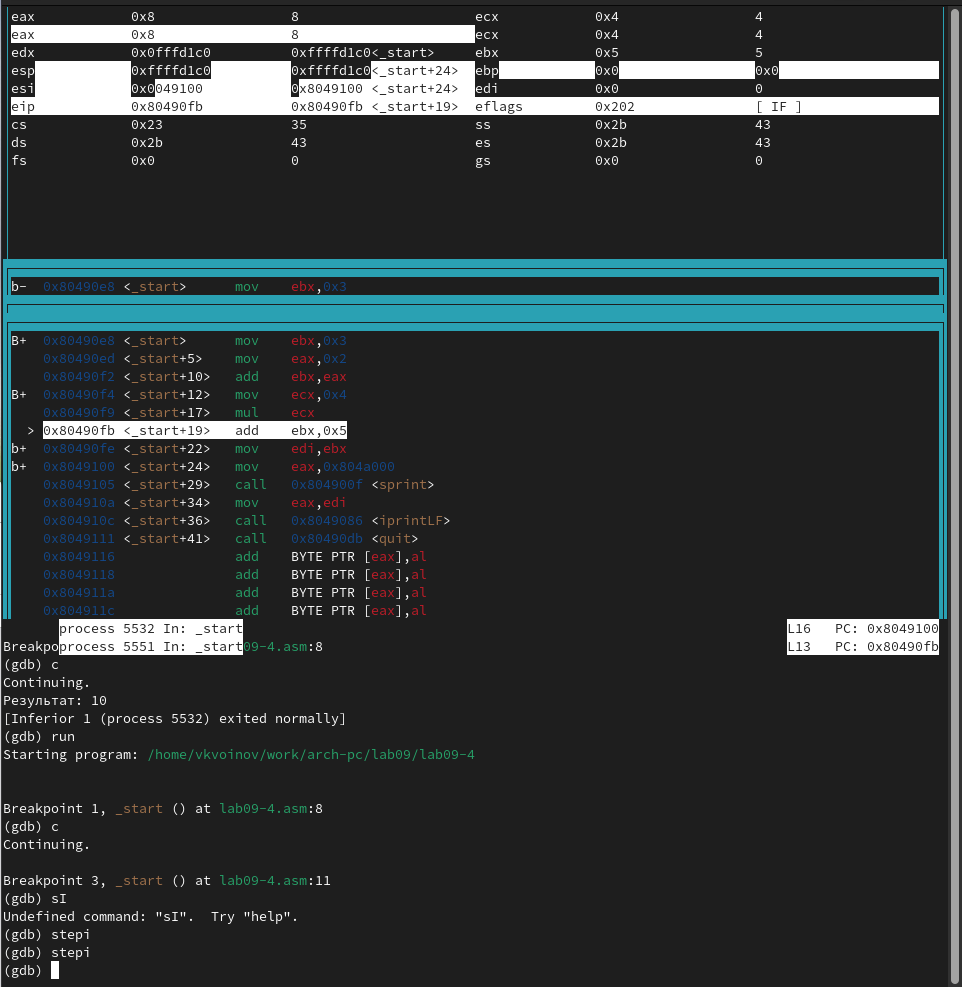


Figure 28: После умножения

Для исправления этой ошибки достаточно поменять местами ebx и eax.(рис. [29](#fig:031)) и (рис. [30](#fig:032)).

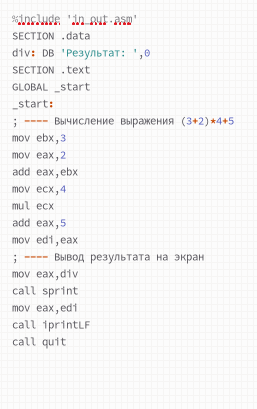


Figure 29: Текст измененной программы

(рис. [30](#fig:032)).

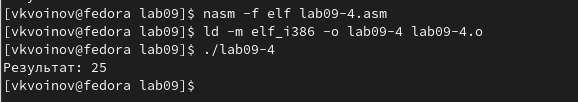


Figure 30: Работа измененной программы

# 4 Выводы

На этой лабораторной работе я приобрел навыки написания программ с использованием подпрограмм, ознакомился с методами отладки при помощи GDB и его основными возможностями.