Отчёт по лабораторной работе № 9

Дисциплина: архитектура компьютера

Мусатова Екатерина Викторовна

Содержание

# 1 Цель работы

Освоить работу с подпрограммами и отладчиком gdb.

# 2 Выполнение лабораторной работы

Создаю необходимый каталог и файлы (рис. [1](#fig:001)).

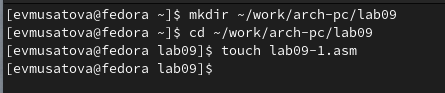


Figure 1: Создание файлов и каталога

Ввожу в файл lab09-1.asm текст программы из листинга 9.1. Создаю исполняемый файл и проверяю его работу (рис. [2](#fig:002)).

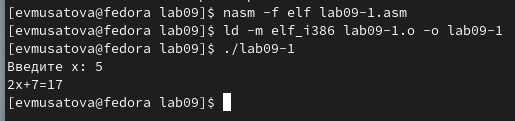


Figure 2: Проверка программы

Добавляю подпрограмму \_subcalcul в подпрограмму \_calcul, для вычисления выражения 𝑓(𝑔(𝑥)), (рис. [3](#fig:003)).

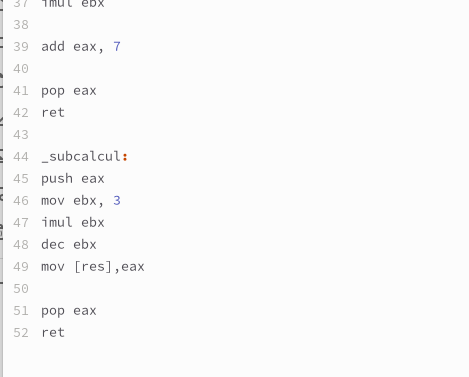


Figure 3: Изменение программы

Создаю исполняемый файл и проверяю работу программы (рис. [4](#fig:004)).

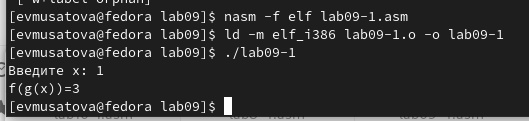


Figure 4: Проверка

Создаю файл lab09-2.asm с текстом программы из Листинга 9.2. Затем получаю исполняемый файл и загружаю его в отладчик gdb (рис. [5](#fig:005)).

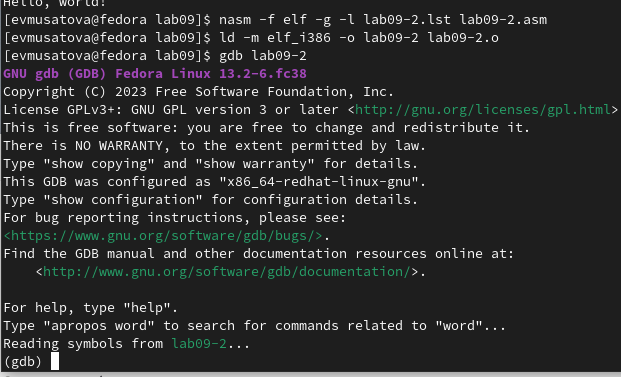


Figure 5: Загрузка в отладчик

Запускакю программу в оболочке GDB с помощью команды run (рис. [6](#fig:006)).

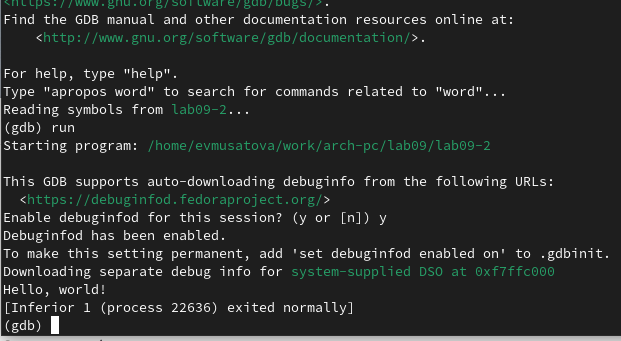


Figure 6: Запуск программы

Для более подробного анализа программы устанавливаю брейкпоинт на метку \_start, с которой начинается выполнение любой ассемблерной программы, и запускаю её (рис. [7](#fig:007)).

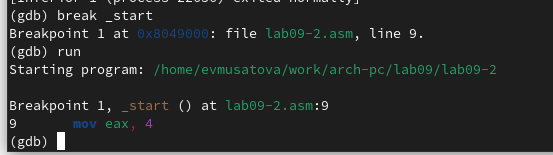


Figure 7: Установка брейкпоинта

Смотрю дисассимилированный код программы с помощью команды disassemble (рис. [8](#fig:008)).

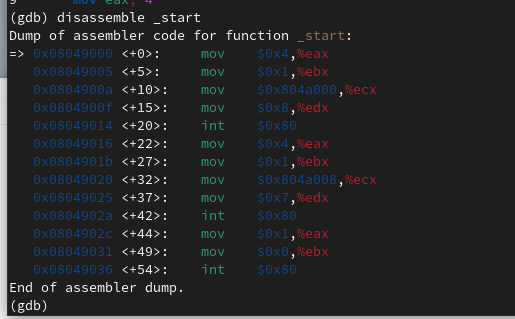


Figure 8: Просмотр кода

Переключаюсь на отображение команд с Intel’овским синтаксисом (рис. [9](#fig:009)). В представлении ATT в виде 16-ричного числа записаны первые аргументы всех комманд, а в представлении intel так записываются адреса вторых аргумантов.

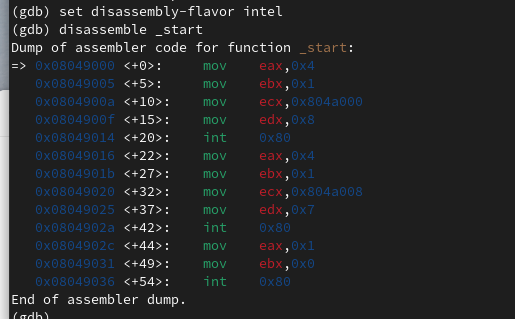


Figure 9: Переключение

Включаю режим псевдографики (рис. [10](#fig:010)).

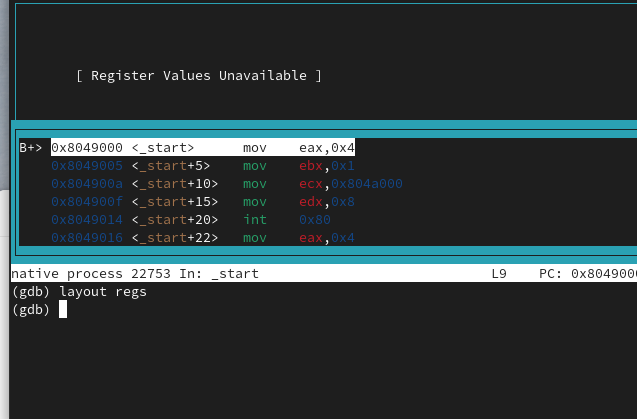


Figure 10: Псевдографика

Проверяю что была установлена точка основа. Затем устанавливаю еще одну точку основа по адресу инструкции (рис. [11](#fig:011)).

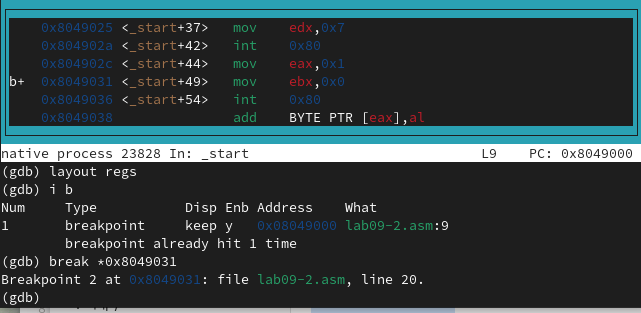


Figure 11: Проверка

Смотрю информацию о всех установленных точках основа (рис. [12](#fig:012)).

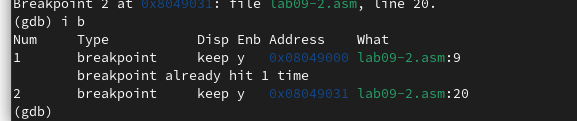


Figure 12: проверка

Выполняю 5 инструкций (рис. [13](#fig:013)).

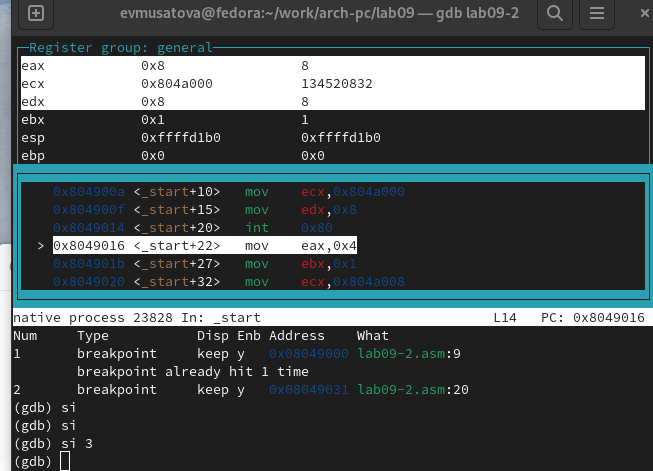


Figure 13: Инструкции

Смотрю значение переменной по имени (рис. [14](#fig:014)).

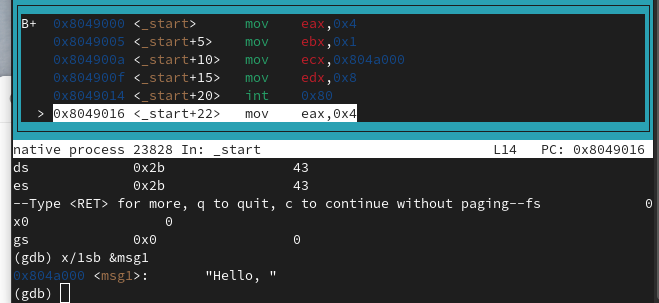


Figure 14: Просмотр

Теперь смотрю значение переменной по адресу (рис. [15](#fig:015)).

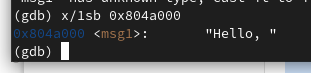


Figure 15: Другой способ посмотреть значение переменной

Изменяю первый символ переменной msg1 (рис. [16](#fig:016)).

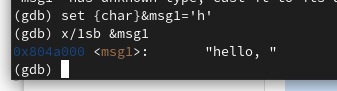


Figure 16: Изменение символа

Теперь заменяю символ во второй переменной (рис. [17](#fig:017)).

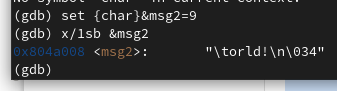


Figure 17: Замена символа в другой преременной

С помощью команды set изменяю значение регистра ebx (рис. [18](#fig:018)). Однако при попытке задать строчное значение, происходит ошибка.

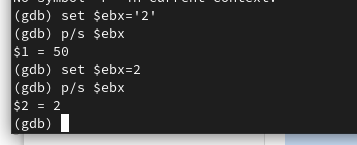


Figure 18: Изменение значения регистра

Завершаю выполнение программы (рис. [19](#fig:019)).



Figure 19: Завершение

Копирую файл lab8-2.asm, созданный при выполнении лабораторной работы №8, создаю исполняемый файл и загружаю его в отладчик, указав аргументы (рис. [20](#fig:020)).

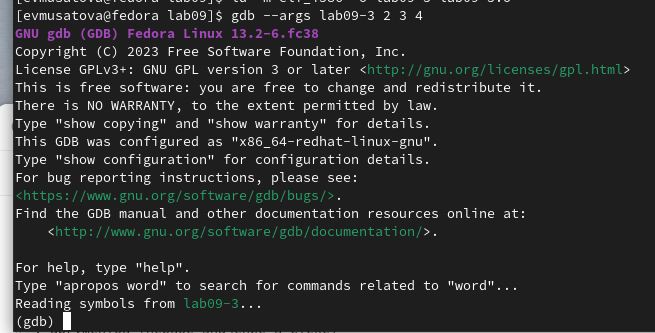


Figure 20: Загрузка в отладчик

Для начала установим точку останова перед первой инструкцией в программе и запустим её (рис. [21](#fig:021)).

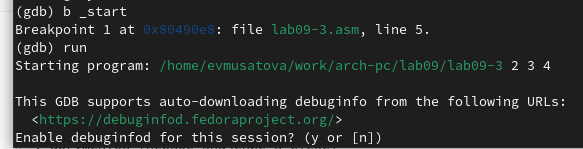


Figure 21: Точка основа и запуск

Смотрим количество аргументов (рис. [22](#fig:022)).

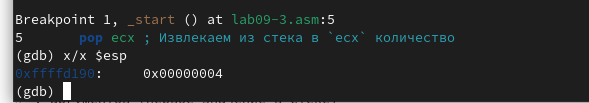


Figure 22: Проверка

Смотрю остальные позиции стека (рис. [23](#fig:023)). Их адреса распологаются в 4 байтах друг от друга(именно столько заниемает элемент стека.

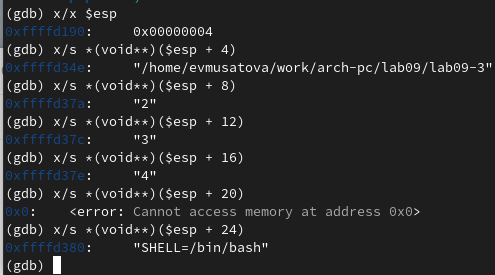


Figure 23: Просмотр

# 3 Выводы

В результате выполнения работы, я научился организовывать код в подпрограммы и познакомился с базовыми функциями отладчика gdb.