Отчёт по лабораторной работе №9

НПМбв-02-21

Гугульян Ксения Александровна

Содержание

# 1 Цель работы

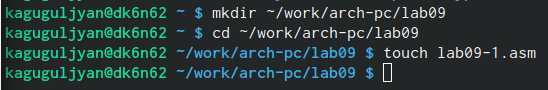
Приобретение навыков написания программ с использованием подпрограмм. Знакомство с методами отладки при помощи GDB и его основными возможностями.

# 2 Задание

1. Создайте каталог для выполнения лабораторной работы № 9, перейдите в него и создайте файл lab09-1.asm.
2. Введите в файл lab09-1.asm текст программы из листинга 9.1. Создайте исполняемый файл и проверьте его работу. Измените 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.
3. Создайте файл lab09-2.asm с текстом программы из Листинга 9.2. Получите исполняемый файл. Загрузите исполняемый файл в отладчик gdb. Проверьте работу программы, запустив ее в оболочке GDB с помощью команды run.
4. Установим брейкпоинт на метку \_start, с которой начинается выполнение любой ассемблерной программы, и запустим её. Посмотрим дисассимилированный код программы с помощью команды disassemble начиная с метки \_start. Переключимся на отображение команд с Intel’овским синтаксисом, введя команду set disassembly-flavor intel. Включаем режим псевдографики для более удобного анализа программы.
5. На предыдущих шагах была установлена точка останова по имени метки (\_start). Проверьте это с помощью команды info breakpoints. Определите адрес предпоследней инструкции (mov ebx,0x0) и установите точку останова. Посмотрите информацию о всех установленных точках останова.
6. Выполните 5 инструкций с помощью команды stepi (или si) и проследите за изменением значений регистров. Значения каких регистров изменяются? Посмотрите значение переменной msg1 по имени. Посмотрите значение переменной msg2 по адресу. Измените первый символ переменной msg1. Замените любой символ во второй переменной msg2. Выведете в различных форматах (в шестнадцатеричном формате, в двоичном формате и в символьном виде) значение регистра edx. С помощью команды set измените значение регистра ebx.
7. Скопируйте файл lab8-2.asm, созданный при выполнении лабораторной работы №8, с программой выводящей на экран аргументы командной строки (Листинг 8.2) в файл с именем lab09-3.asm. Создайте исполняемый файл. Загрузите исполняемый файл в отладчик, указав аргументы. Исследуем расположение аргументов командной строки в стеке после запуска программы с помощью gdb.

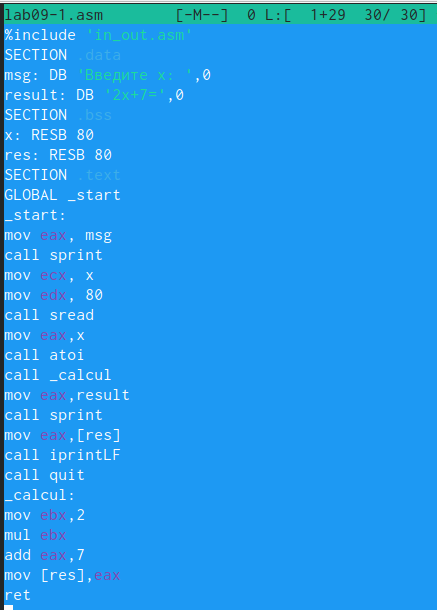
# 3 Выполнение лабораторной работы

1. Создаём каталог для выполнения лабораторной работы № 9, перейдём в него и создаём файл lab09-1.asm (рис. ??).



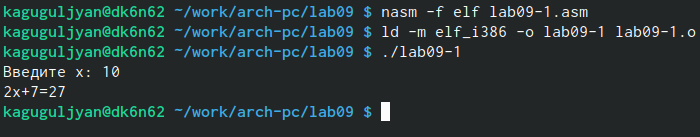
Создание каталога

1. Введём в файл lab09-1.asm текст программы из листинга 9.1 (рис. ??).



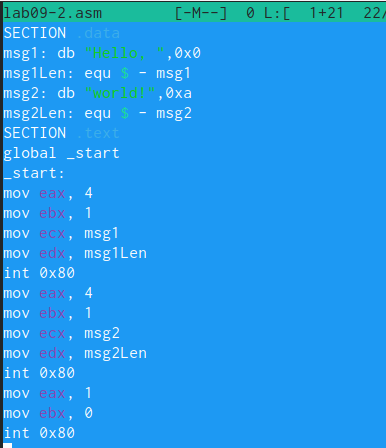
Ввод текста в файл

Создаём исполняемый файл и проверяем его работу (рис. ??).



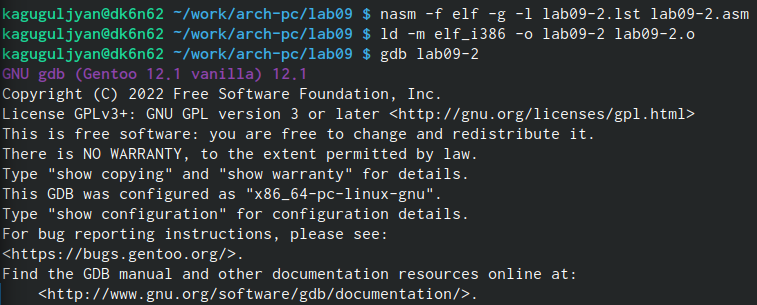
Создание исп. файла

1. Создаём файл lab09-2.asm с текстом программы из Листинга 9.2 (рис. ??).



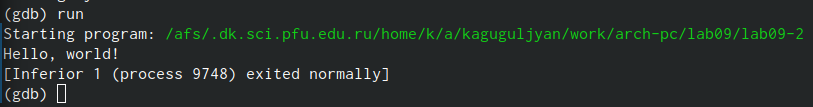
Создание файла с текстом

Получаем исполняемый файл и загружаем исполняемый файл в отладчик gdb (рис. ??).



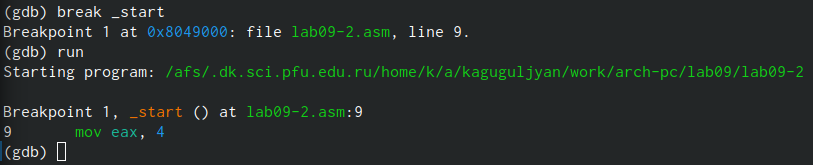
Загрузка исп. файла в отладчик

Проверяем работу программы, запустив ее в оболочке GDB с помощью команды run (рис. ??).



Проверка

1. Установим брейкпоинт на метку \_start, с которой начинается выполнение любой ассемблерной программы, и запустим её (рис. ??).



Установка брейкпоинт

Посмотрим дисассимилированный код программы с помощью команды disassemble начиная с метки \_start (рис. ??).



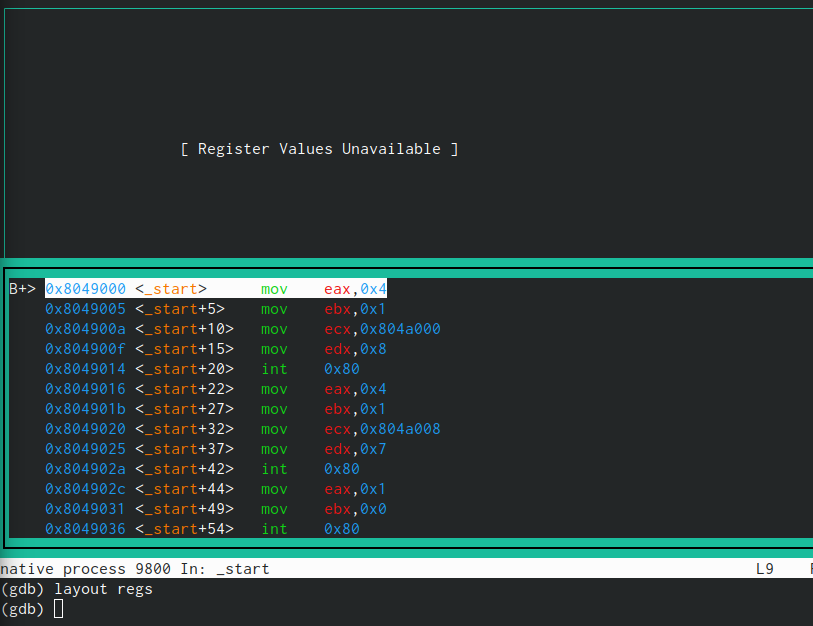
Просмотр программы

Переключимся на отображение команд с Intel’овским синтаксисом, введя команду set disassembly-flavor intel (рис. ??).



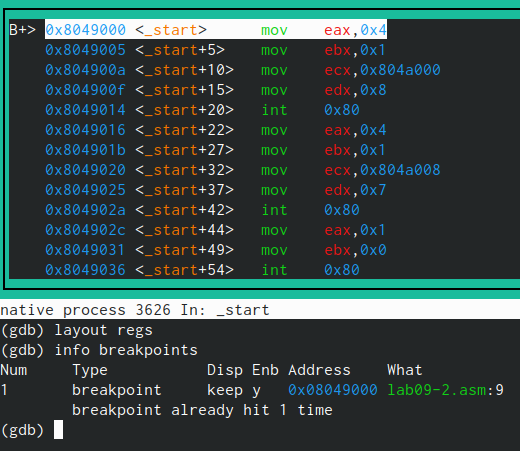
Переключение

Включаем режим псевдографики для более удобного анализа программы (рис. ??).



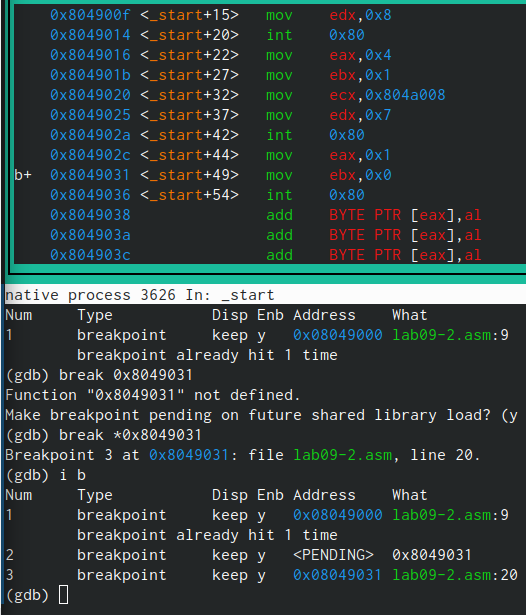
Включение режима

1. Проверим это с помощью команды info breakpoints (рис. ??).



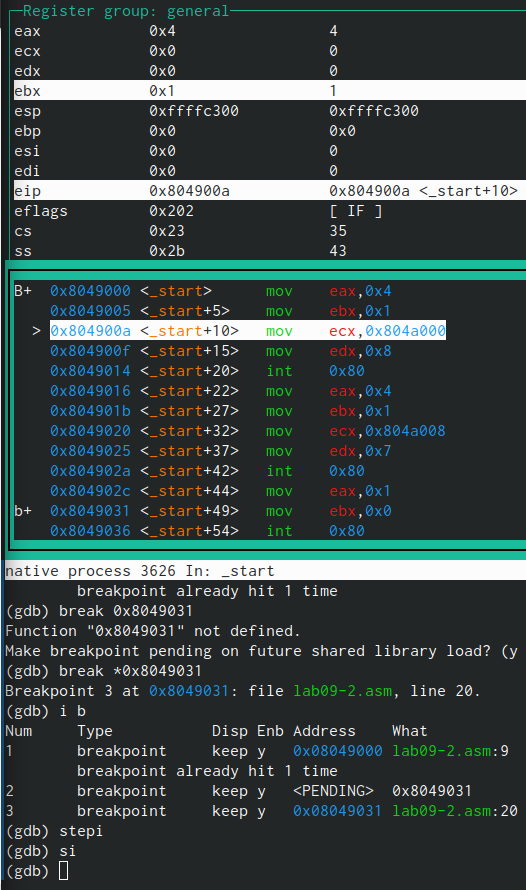
Команда info breakpoints

Определим адрес предпоследней инструкции (mov ebx,0x0) и установим точку останова. Посмотрим информацию о всех установленных точках останова (рис. ??).



Определение адреса и просмотр информации

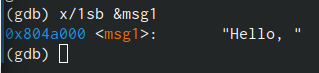
1. Выполним 5 инструкций с помощью команды stepi (или si) и проследим за изменением значений регистров (рис. ??).



Инструкции с помощью stepi

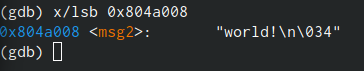
Меняются значения регистров ecx.

Посмотрим значение переменной msg1 по имени (рис. ??).



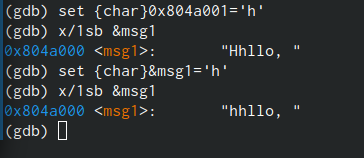
Просмотр значения переменной msg1 по имени

Посмотрим значение переменной msg2 по адресу (рис. ??).



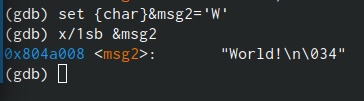
Просмотр значения перемнной msg2 по адресу

Изменим первый символ переменной msg1 (рис. ??).



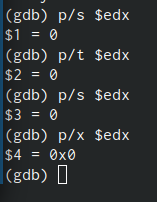
Изменение символов перемнной msg1

Заменим любой символ во второй переменной msg2 (рис. ??).



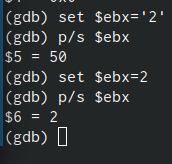
Замена символов перемнной msg2

Выведем в различных форматах (в шестнадцатеричном формате, в двоичном формате и в символьном виде) значение регистра edx (рис. ??).



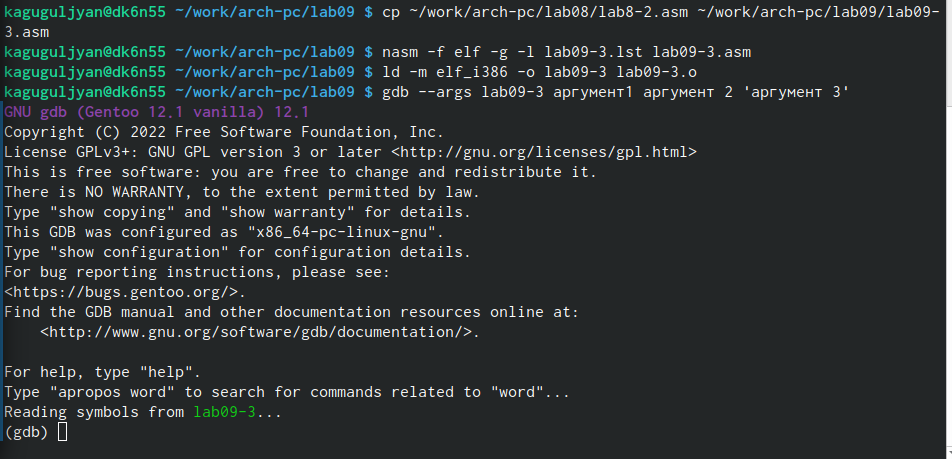
Ввод в различных форматах регистра edx

С помощью команды set изменим значение регистра ebx (рис. ??).



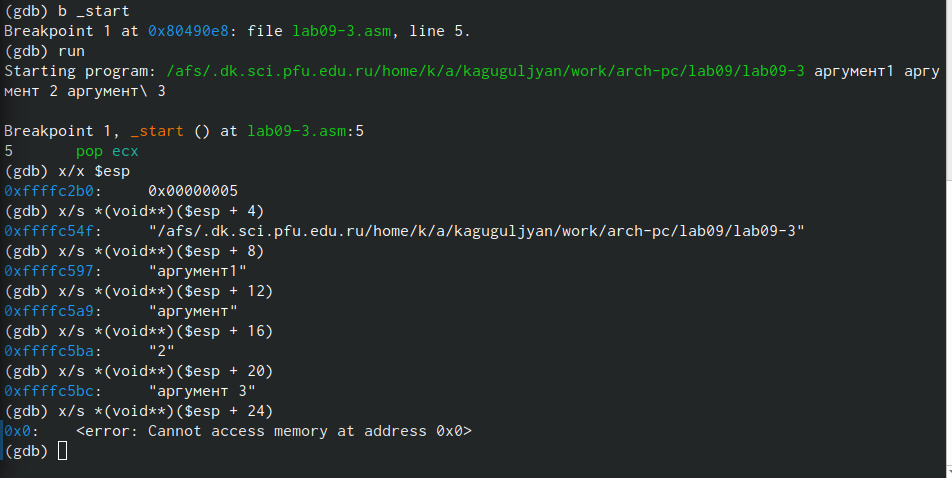
Изменение регистра ebx

1. Скопируем файл lab8-2.asm, созданный при выполнении лабораторной работы №8, с программой выводящей на экран аргументы командной строки (Листинг 8.2) в файл с именем lab09-3.asm. Создаём исполняемый файл. Загружаем исполняемый файл в отладчик, указав аргументы (рис. ??).



Копирование файла, создание исп. файла и загрузка

Исследуем расположение аргументов командной строки в стеке после запуска программы с помощью gdb (рис. ??).



Исследование расположения аргументов к.с. в стеке

# 4 Выводы

В ходе лабораторной работы я приобрела навыки написания программ с использованием подпрограмм. Ознакомилась с методами отладки при помощи GDB и его основными возможностями.

# Список литературы