Отчёт по лабораторной работе №10

Дисциплина: ‘архитектура компьютеров’

Бабенко Роман Игоревич

Содержание

# 1 Цель работы

Приобрести навыки программирования с использованием подпрограмм и познакомиться с методом отладки ‘GDB’

# 2 Выполнение лабораторной работы

Создаём каталог для выполнения лабораторной работы, переходим в него и создаём файл lab10-1.asm (рис. 1)

Рис. 1: Создание каталога и файла

Рис. 1: Создание каталога и файла

Вводим в файл программу из листинга, создаем исполняемый файл и проверяем его работу (рис. 2) и (рис. 3)

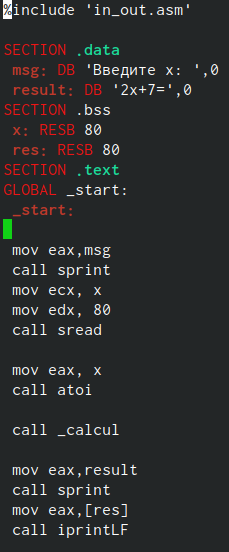


Рис. 2: Вводим программу в файл

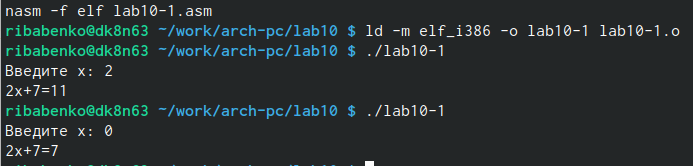


Рис. 3: Проверка работы файла

Добавляем подпрограмму ’\_subcalcul’(рис. 4)

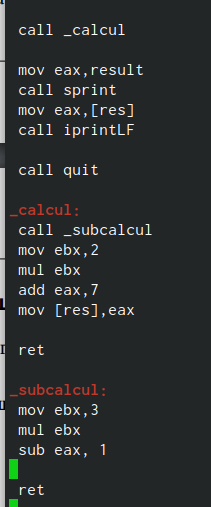


Рис. 4: Добавляем подпрограмму

Проверяем изменённую программу (рис. 5)

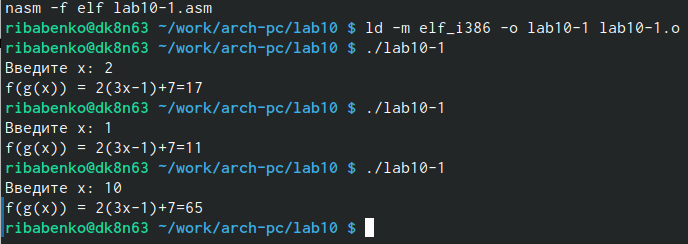


Рис. 5: Проверка программы

Создаём файл lab10-2.asm и записываем в ного програму из листинга 10.2 (рис. 6)

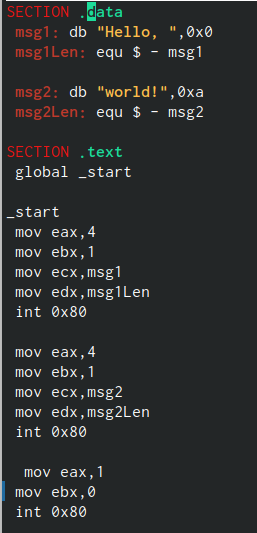


Рис. 6: Записываем программу из листинга

Загружаем исполняемый файл в отладчик gdb (рис. 7)

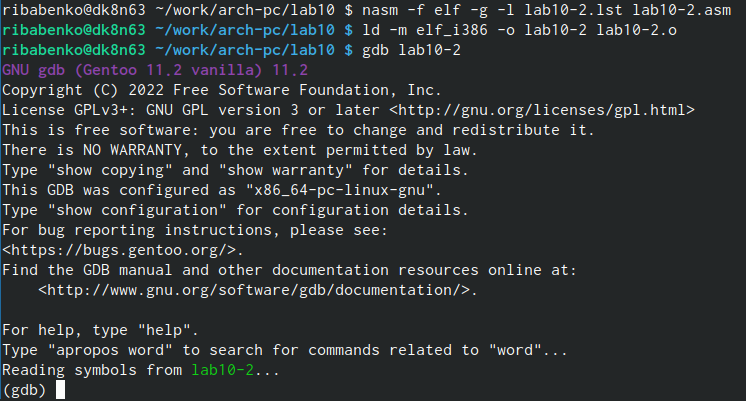


Рис. 7: Используем отладчик

Запускаем программу с помощью команды run (рис. 8)

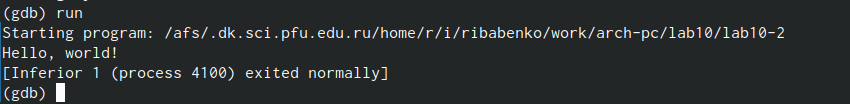


Рис. 8: Запуск программы в отладчике

Устанавливаем брейкпоинт на метку \_start и далее запускаем её (рис. 9)

Рис. 9: Установка брейкпоинта

Рис. 9: Установка брейкпоинта

Смотрим дисассимилированнйы код с помощью программы disassemble начиная с метки \_start (рис. 10)

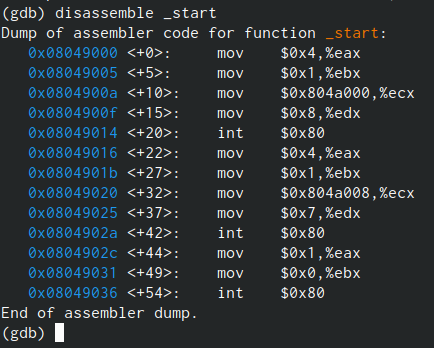


Рис. 10: Просмотр дисассимилированного кода

Переключаемся на отображение команд с Intel’овским синтаксисом, введя команду ‘set disassembly-flavor intel’. С интеловском синтаксе, в отличие от стандартного, в третьем столбце сначала пишется название регистра(рис. 11)

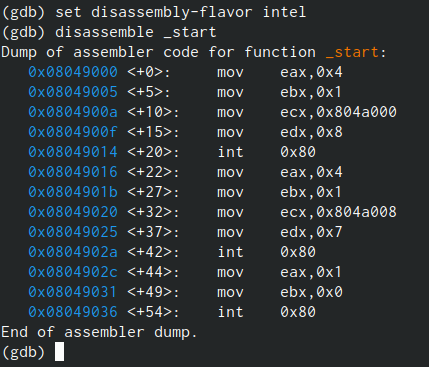


Рис. 11: Переключение синтаксиса

Включаем режим псевдографики для более удобного анализа программы (рис. 12)

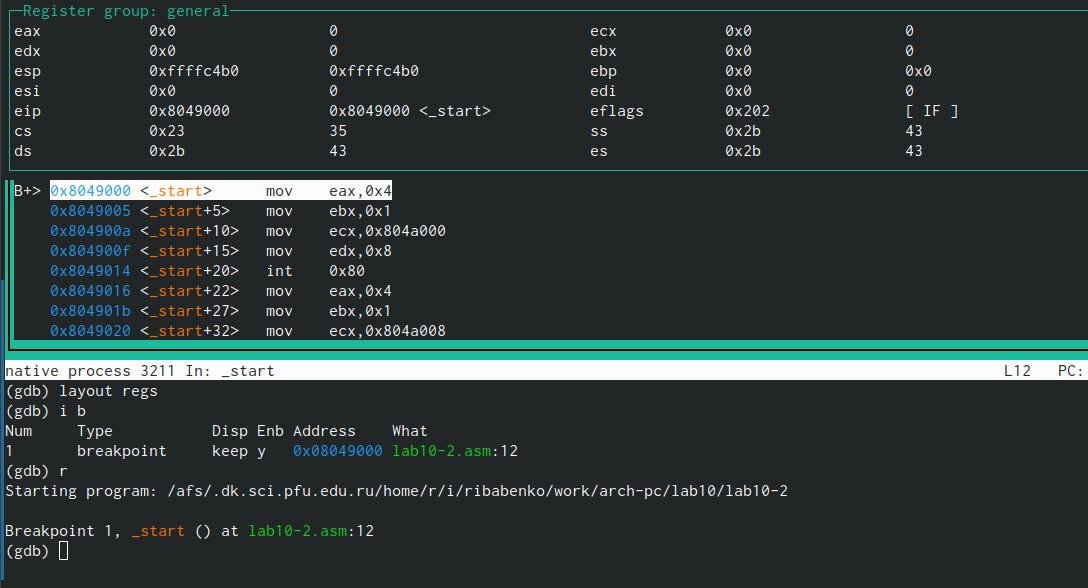


Рис. 12: Включение режима псевдографики

Проверяем наличие точки останова с помощью команды info breakpoints (рис. 13)

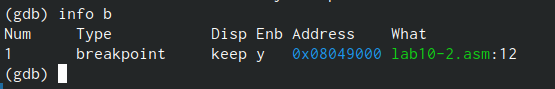


Рис. 13: Проверка точки останова

Устанавливаем ещё одну точку установа по адресу инструкции и смотрим информауию о всех точках останова(рис. 14)

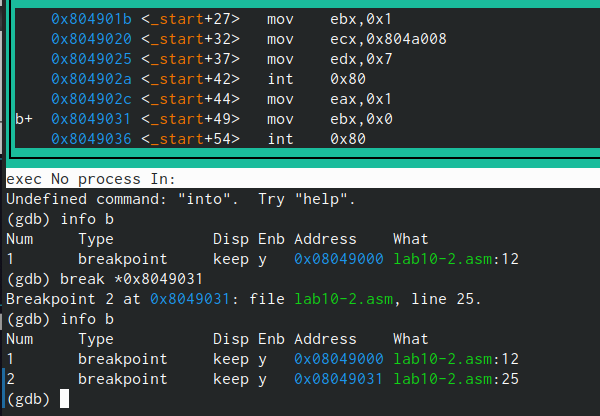


Рис. 14: Установка второй точки останова

Просматриваем содержимое регистров с помощью команды info registers (рис. 15)

Рис. 15: Просмотр содержимого регистров

Рис. 15: Просмотр содержимого регистров

Просматриваем значение переменной msg1 по имени (рис. 16)

Рис. 16: Просмотр значения msg1

Рис. 16: Просмотр значения msg1

Просматриваем значение переменной msg2 по адресу(рис. 17)

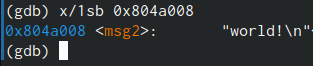


Рис. 17: Просмотр значения msg2

Изменим первый символ переменной msg1 с помощью команды set(рис. 18)

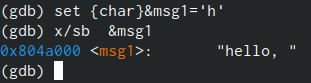


Рис. 18: Изменяем символ в переменной msg1

Также заменим символ в перменной msg2(рис. 19)

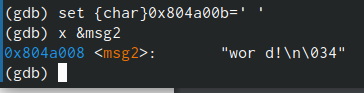


Рис. 19: Изменим символ в msg2

Выведем значения регистра edx в разных форматах(рис. 20)

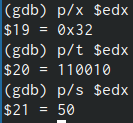


Рис. 20: Вывод значения регистра edx

С помощью команды set изменим значение регистра ebx. C p/s выводит значение в символьном виде(рис. 21)

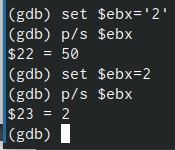


Рис. 21: Изменяем значение ebx

Завершаем выполнение программы с помощью ‘countinue’(рис. 22)

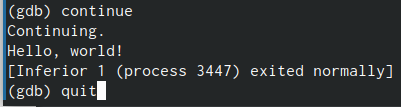


Рис. 22: Завершаем выполнение программы

Копируем файл созданный при выполнении лабораторной работы номер девять и создайм исполняемый файл(рис. 23)

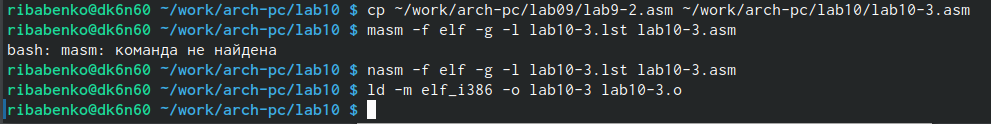


Рис. 23: Создание исполняемого файла для программы из девятой лабораторной работы

Загружаем в gdb программу с аргументами используя ключ –args(рис. 24)

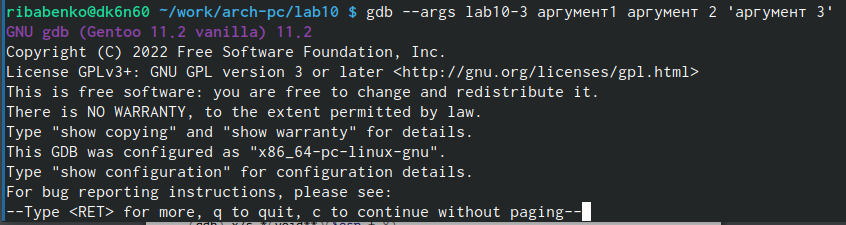


Рис. 24: Зазружаем в gdb программу с агрументами

Устанавливаем точку останова перед первой инструкцией и запускаем её(рис. 25)

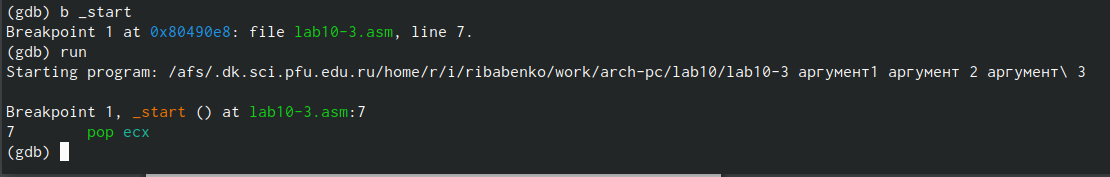


Рис. 25: Делаем точку останова и запускаем инструкцию

Посмотрим различные позиции стека(рис. 26)

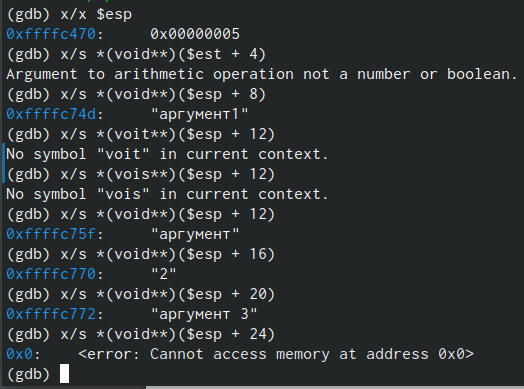


Рис. 26: Смотрим позиции стека

#Задания для самостоятельной работы

Напишем программу из лабораторной работы номер девять и реальзуем в ней вычисление функции через подпрограмму(рис. 27)

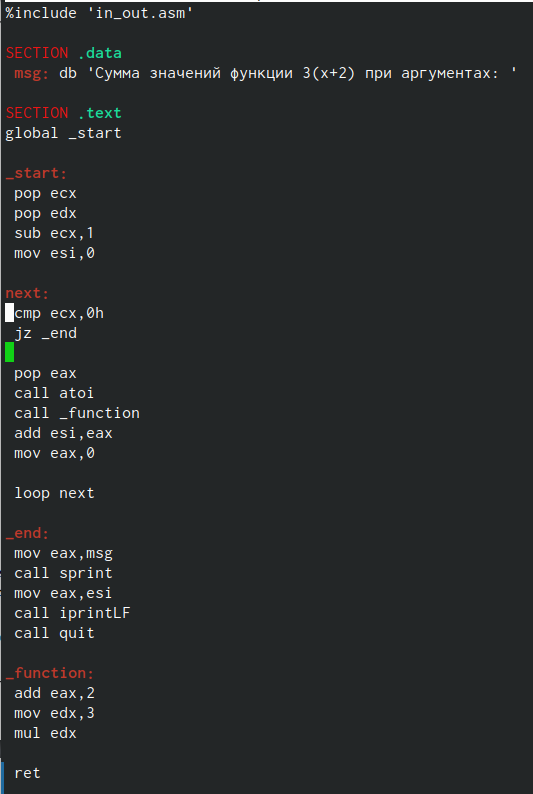


Рис. 27: Написанная программа

Проверяем нашу программу(рис. 28)

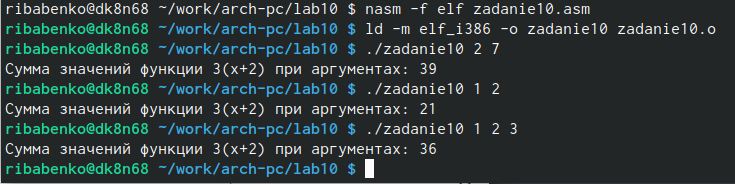


Рис. 28: Проверка программы

Испровляем программу для вычилсения умножения, поменяв регистры местами (там умножался на 4 не регистр ebx, f htubcnh eax)(рис. 29)

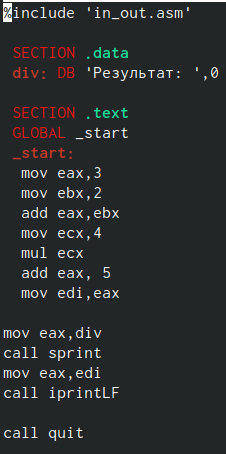


Рис. 29: Исправленная программа

Проверяем исправленную программу(рис. 30)

Рис. 30: Полученный результат

Рис. 30: Полученный результат

# 3 Выводы

В ходе выполнения данной лабораторной работы я приобрёл навыки программирования с использованием подпрограмм и познакомился с методом отладки ‘GDB’