Лабораторная работа №9

Понятие подпрограммы. Откладчик GDB.

Славинский Владислав Вадимович

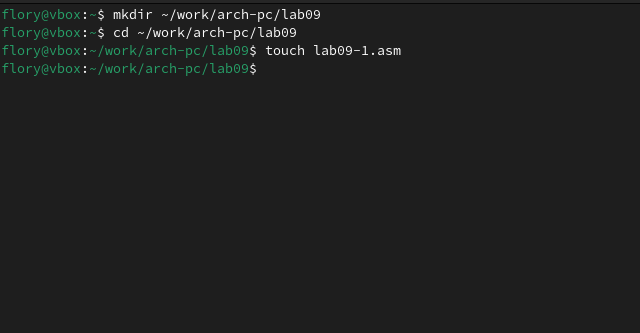
Содержание

# 1 Цель работы

Приобрести навыки написания программ с использованием подпрограмм и ознакомиться с методами отладки при помощи GDB и его основными возможностями.

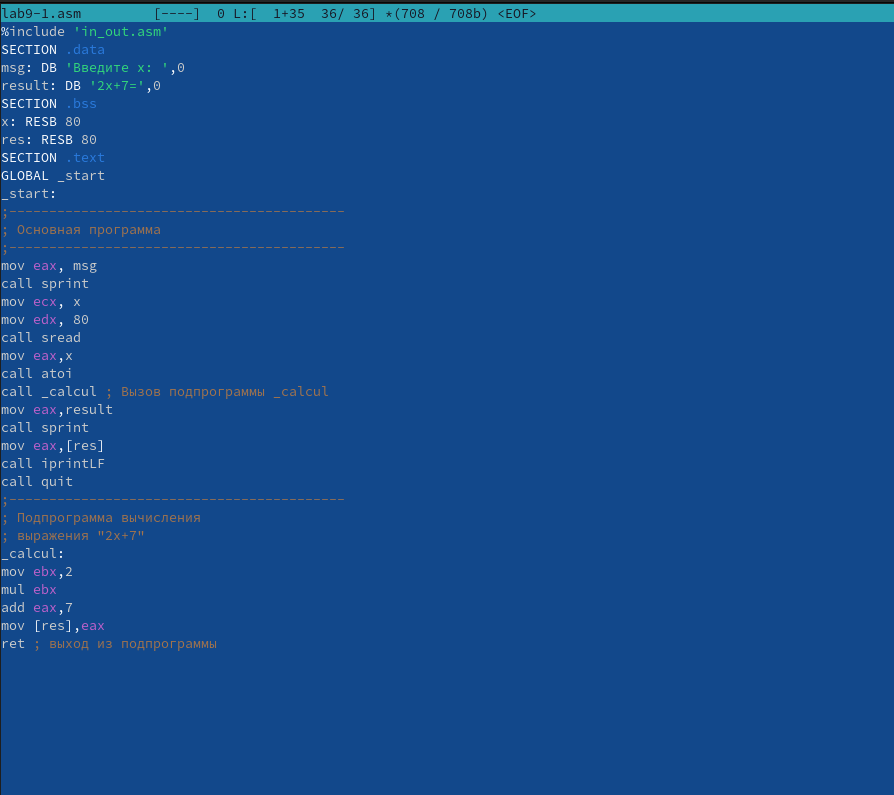
# 2 Выполнение лабораторной работы

1)Создал каталог для лабораторной работы №9 и создал файл lab9-1.asm: (Рис. 2.1)



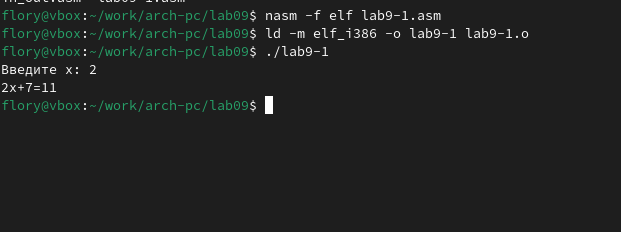
Создание lab9-1

2)Ввел код программы с использованием вызова подпрограммы: (Рис. 2.2)



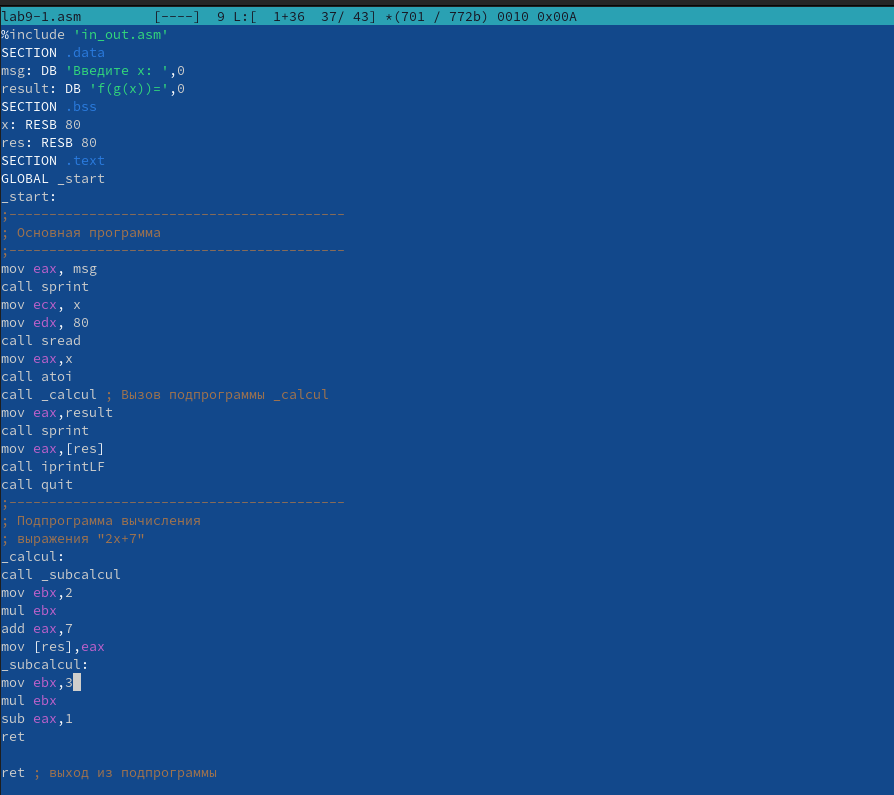
Код программы

3)Запуск программы: (Рис. 2.3)



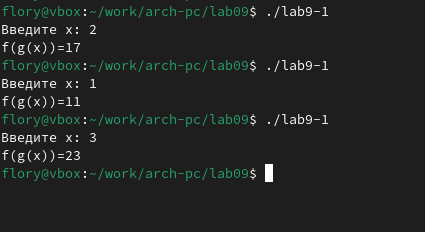
Запуск

4)Теперь изменю файл так, чтобы внутри подпрограммы была подпрограмма, которая вычисляет значение g(x): (Рис. 2.4)



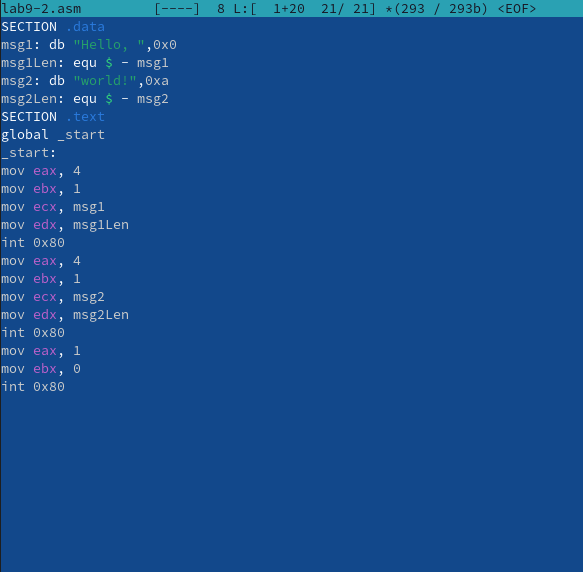
Замена

5)Запустил измененный файл: (Рис. 2.5)



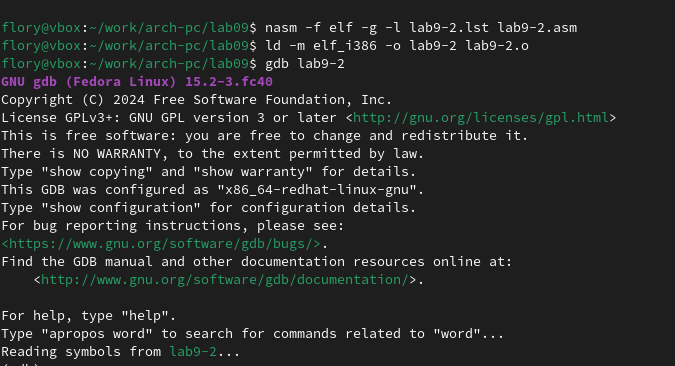
Запуск измененной программы

6)Теперь создам файл lab09-2.asm с текстом программы из Листинга 9.2 и заполню его: (Рис. 2.6)



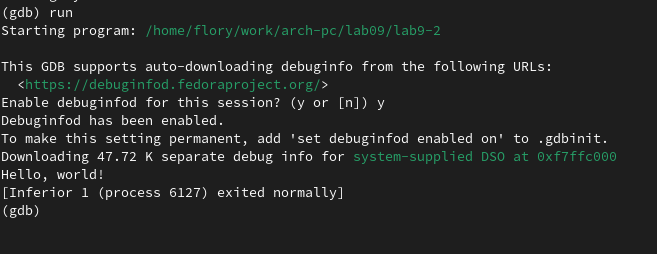
Код программы

7)Запустим программу и использованием -g:(Рис. 2.7)



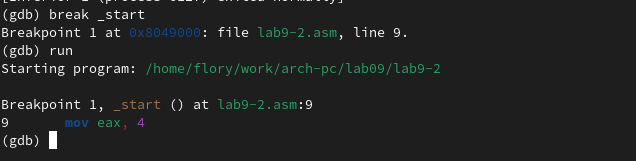
Программа

8)Теперь запустим в отладчике при помощи команды run:(Рис. 2.8)



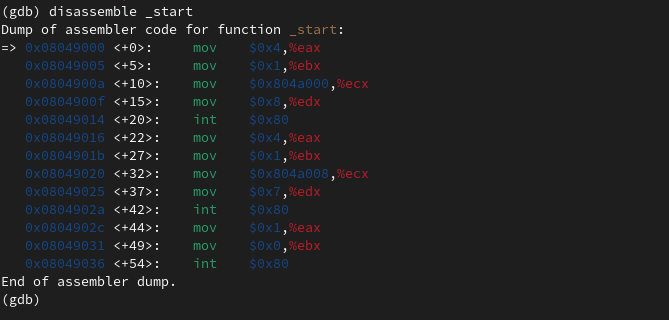
Запуск

9)Создам брейкпоинт на метке \_start с помощью break:(Рис. 2.9)



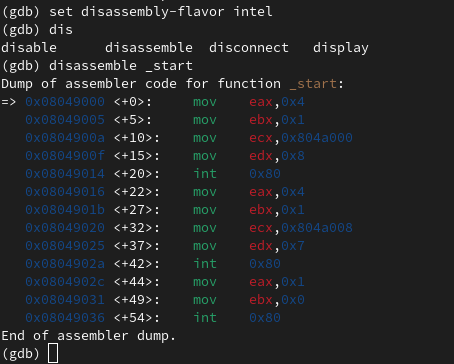
Брейкпоинт

10)Дизассемблируем её:(Рис. 2.10)



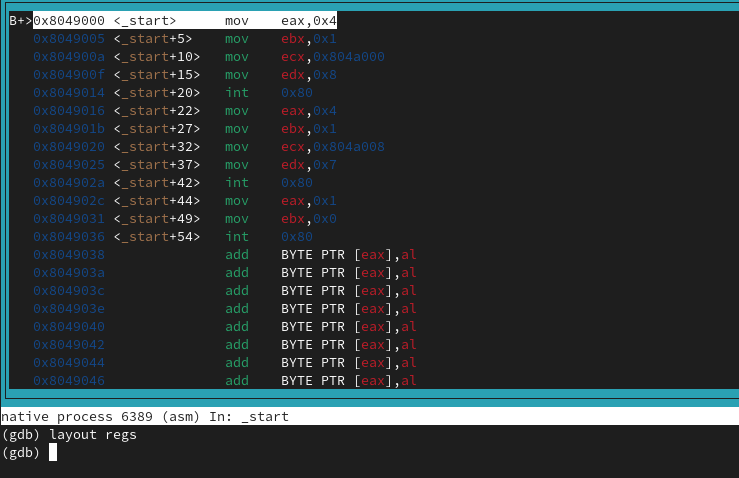
Дизассемблирование

11)Теперь переключусь на отображение команд с интеловским синтаксисом с помощью команды set disassembly-flavor intel и снова дизассемблировал:(Рис. 2.11)



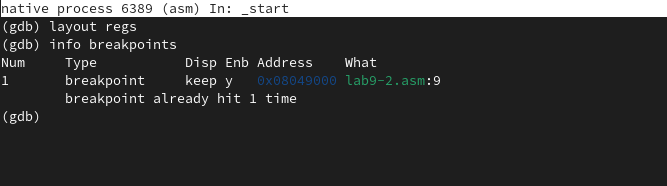
Переключение на другой синтаксис

12)Включу режим псевдографики:(Рис. 2.12)



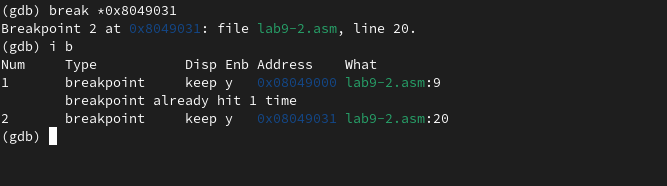
Включение режима

13)Информация о брейкпоинтах:(Рис. 2.13)



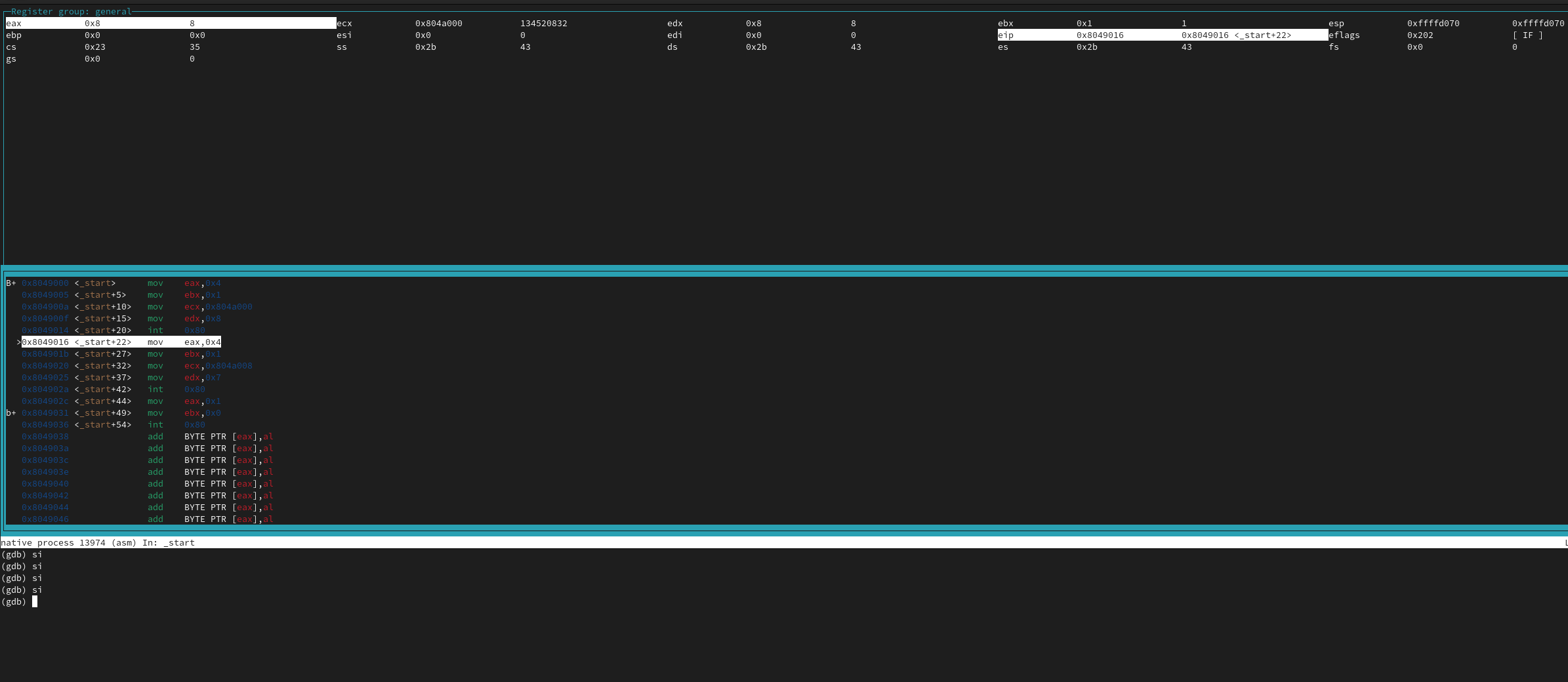
Вывод информации

14)Создам брейпоинт по адресу выведу информацию о брейкпоинтах:(Рис. 2.14)



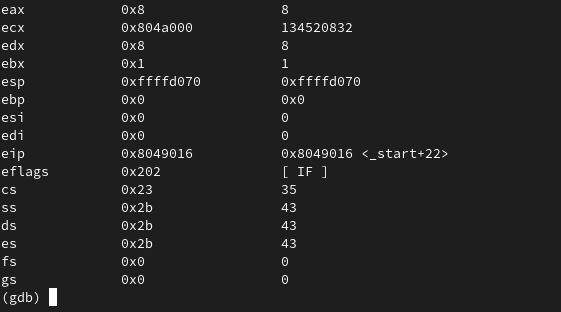
Создание и информация о брейкпоинтах

15)После того, как прописал 5 раз команду si, поменялись значения регистров eax, ecx, edx и ebx:(Рис. 2.15)



si

16)Выведу информацию о значениях регистров:(Рис. 2.16)



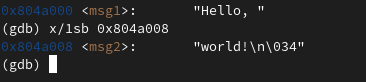
Информация о регистрах

17)Теперь посмотрю значение переменной msg1 по имени:(Рис. 2.17)

Вывод значения msg1

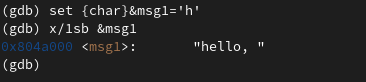
Вывод значения msg1

18)Теперь посмотрю значение переменной msg2 по адресу:(Рис. 2.18)



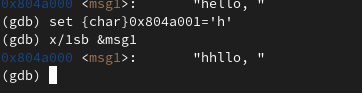
Вывод значения msg2

19)Попробую изменить первый символ переменной:(Рис. 2.19)



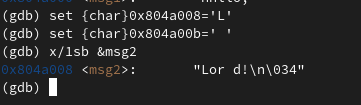
Смена первого символа

20)Изменю второй символ переменной, обратясь по адресу:(Рис. 2.20)



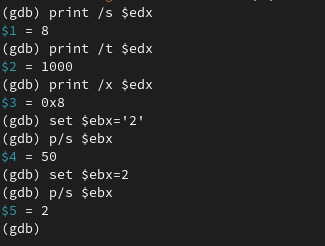
Смена первого символа

21)Изменю несколько символов второй переменной:(Рис. 2.19)



Смена первого символа

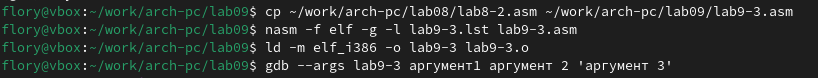
22)Выведу значения регистра в строковом, двоичном и шестнадцатеричном виде и изменю значения регистра:(Рис. 2.22)



Смена первого символа

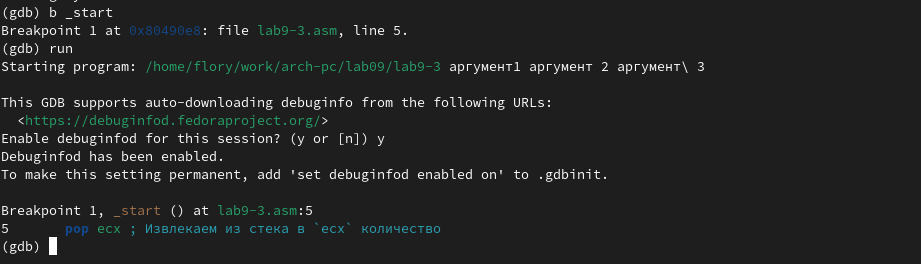
В регистр записались разные значения. Всё из-за того, что мы пишем в одном случае число, а в другом строку.

23)Скопирую файл lab8-2.asm в файл lab9-3.asm и создам исполняемый файл: (Рис. 2.23)



Копирование

24)Создам брейкпоинт и запущу программу: (Рис. 2.23)



Запуск

25)Теперь выведу значение регистра esp и выведу значение всех элементов: (Рис. 2.23)

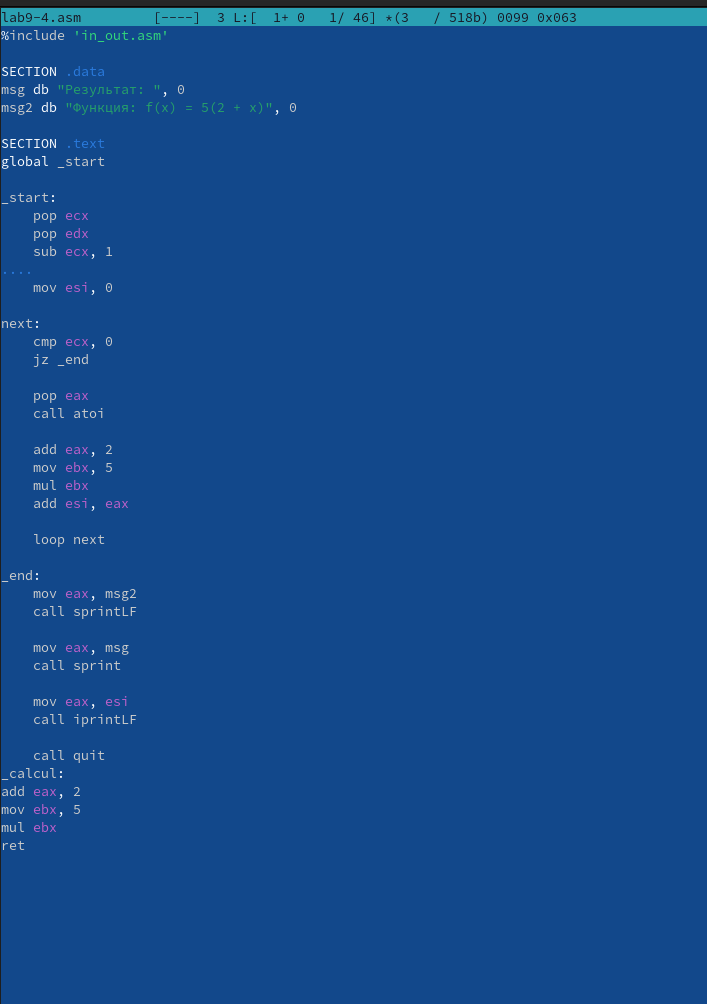


Вывод

Из-за того, что с шагом 4 располагаются данные в стеке, для каждого элемента нужно менять значение адреса с шагом 4.

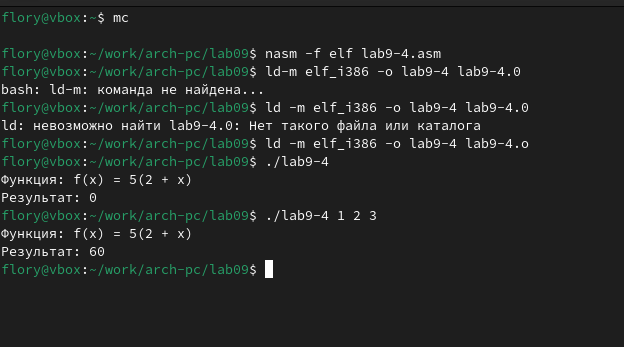
# 3 Выполнение задания для самостоятельной работы

1)Скопировал первый файл самостоятельной работы из лабораторной работы №8 и отредактировал код :(Рис. 3.1)



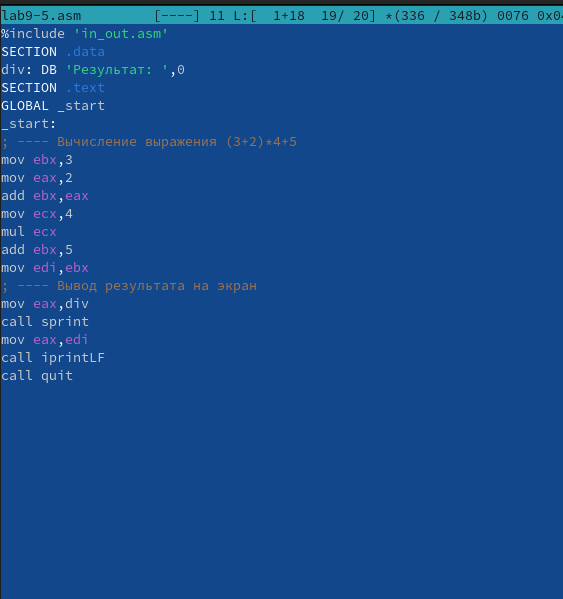
Создание lab9-4.asm

2)Запуск программы:(Рис. 3.2)



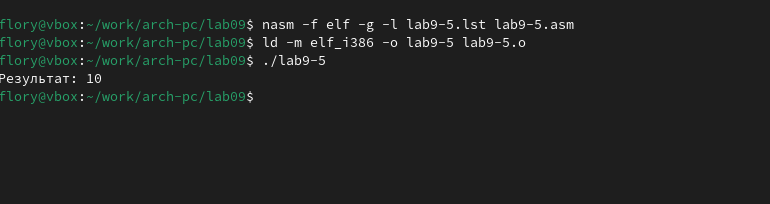
Запуск

3)Создам теперь файл для второй самостоятельной работы и вставлю код из листинга 9.3:(Рис. 3.3)



Код

4)Соберем программу и запустим её. Видим, что ответ не правильный:(Рис. 3.4)



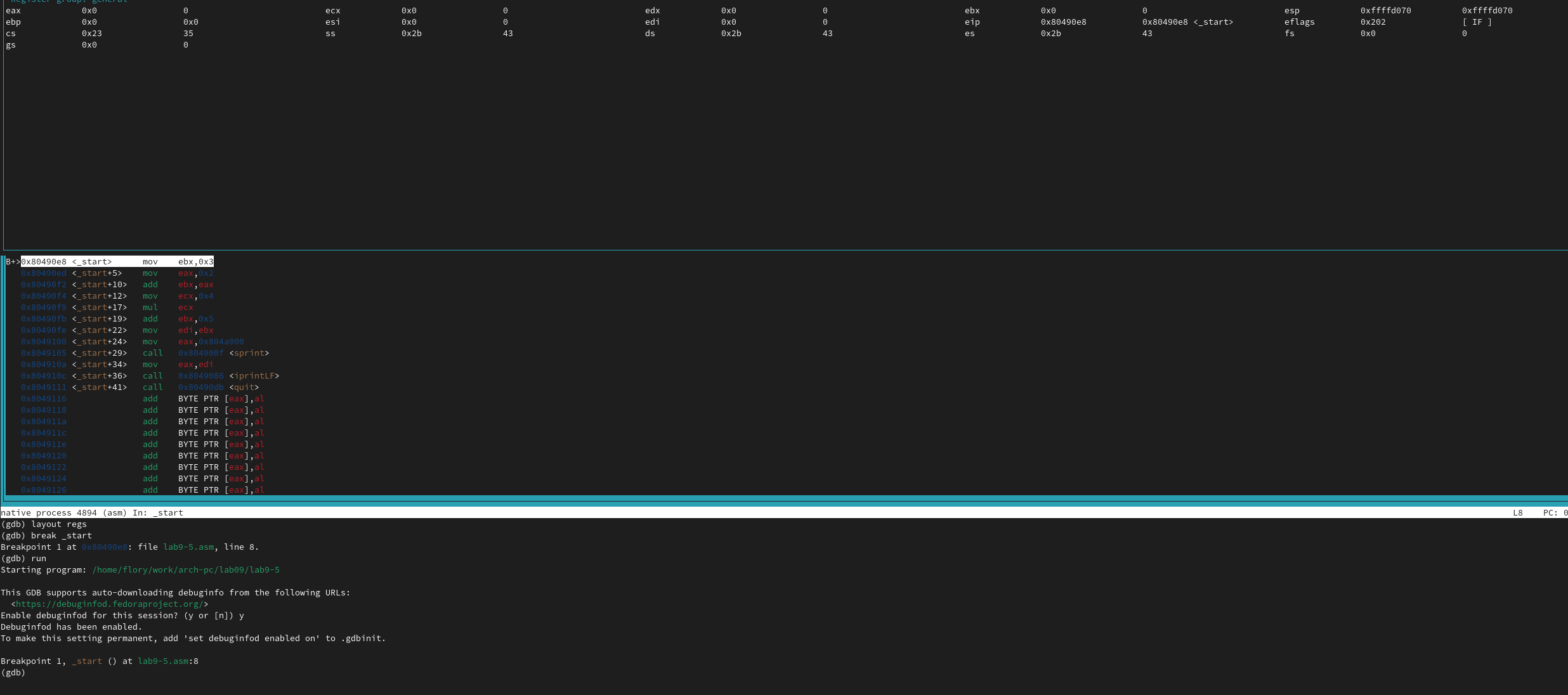
Запуск

1. Перейду в gdb и установлю брейкпоинт на \_start:(Рис. 3.5)



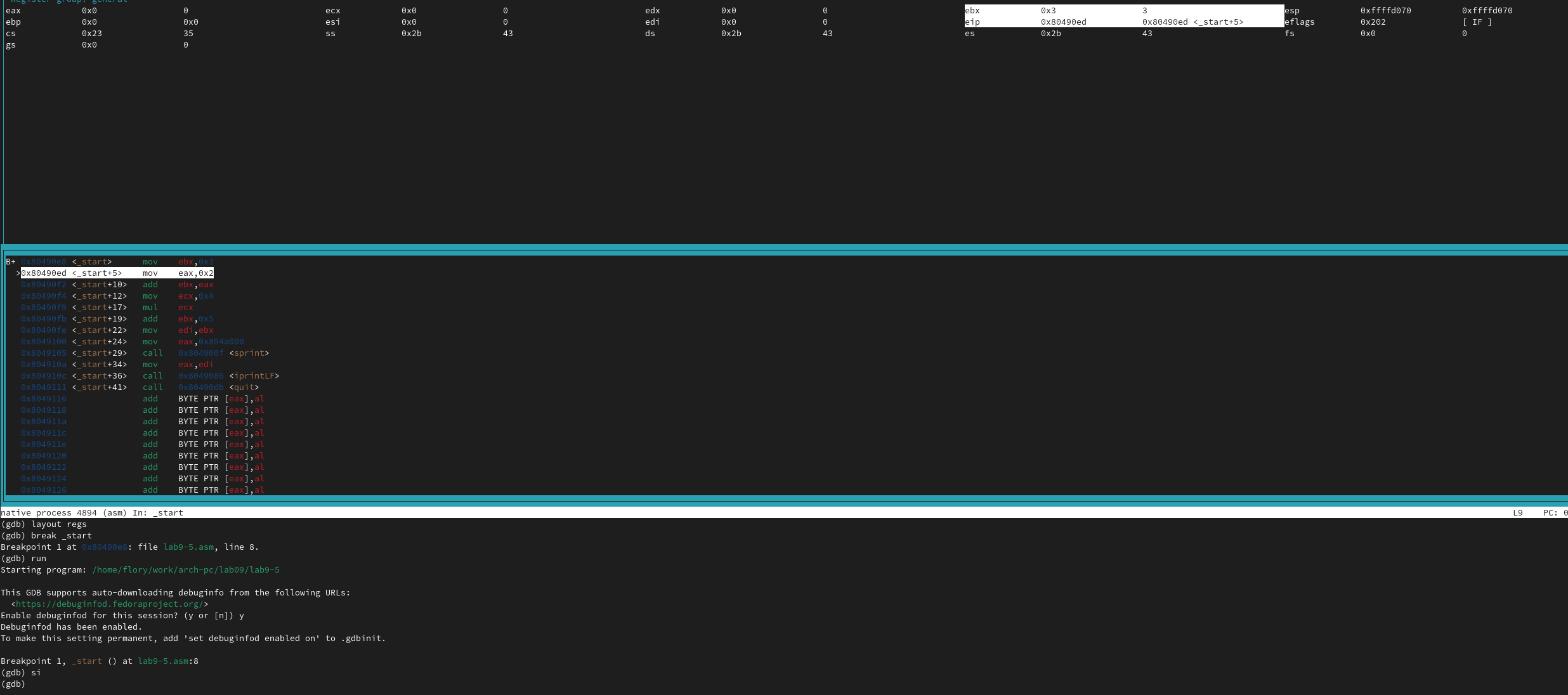
Установка брейкпоинта

1. Значение регистров на 0 шаге:(Рис. 3.6)



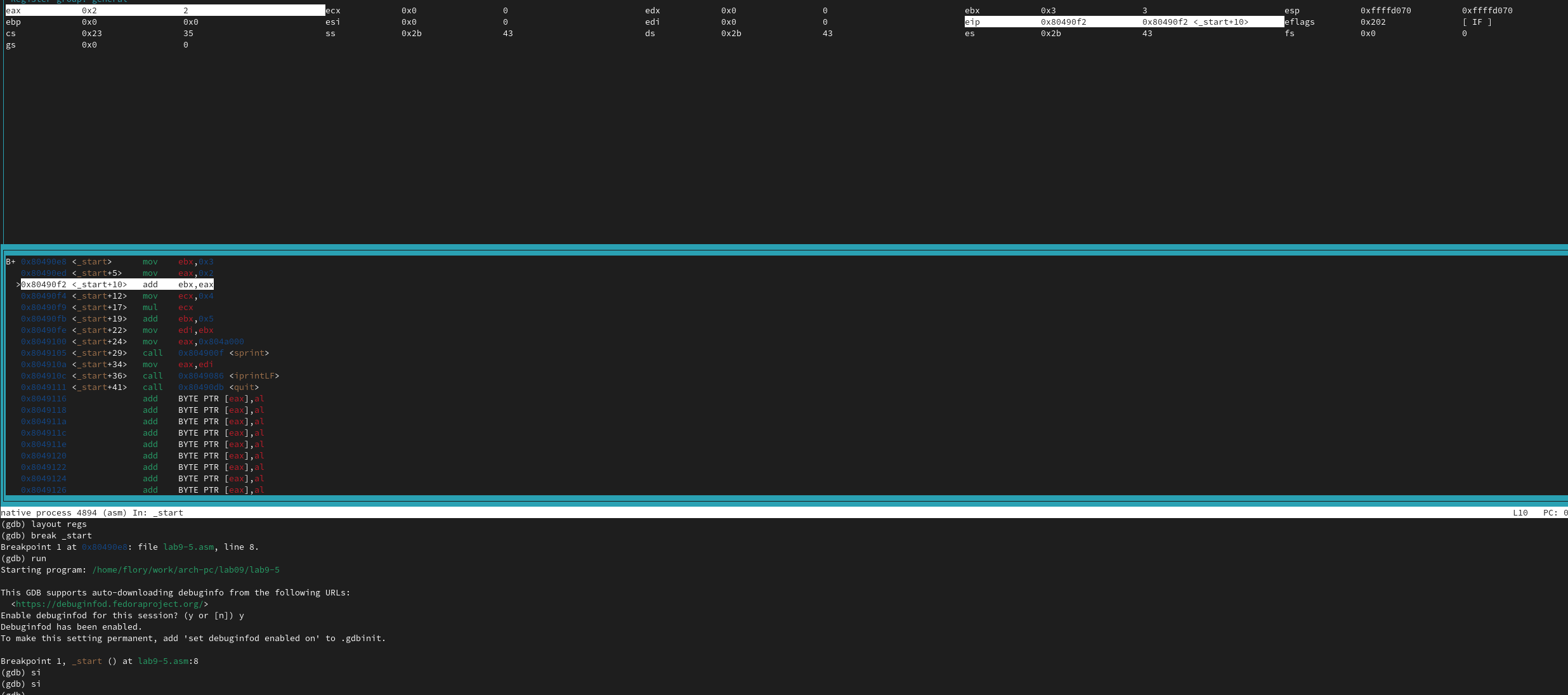
Значение регистров

1. Значение регистров на 1 шаге:(Рис. 3.7)



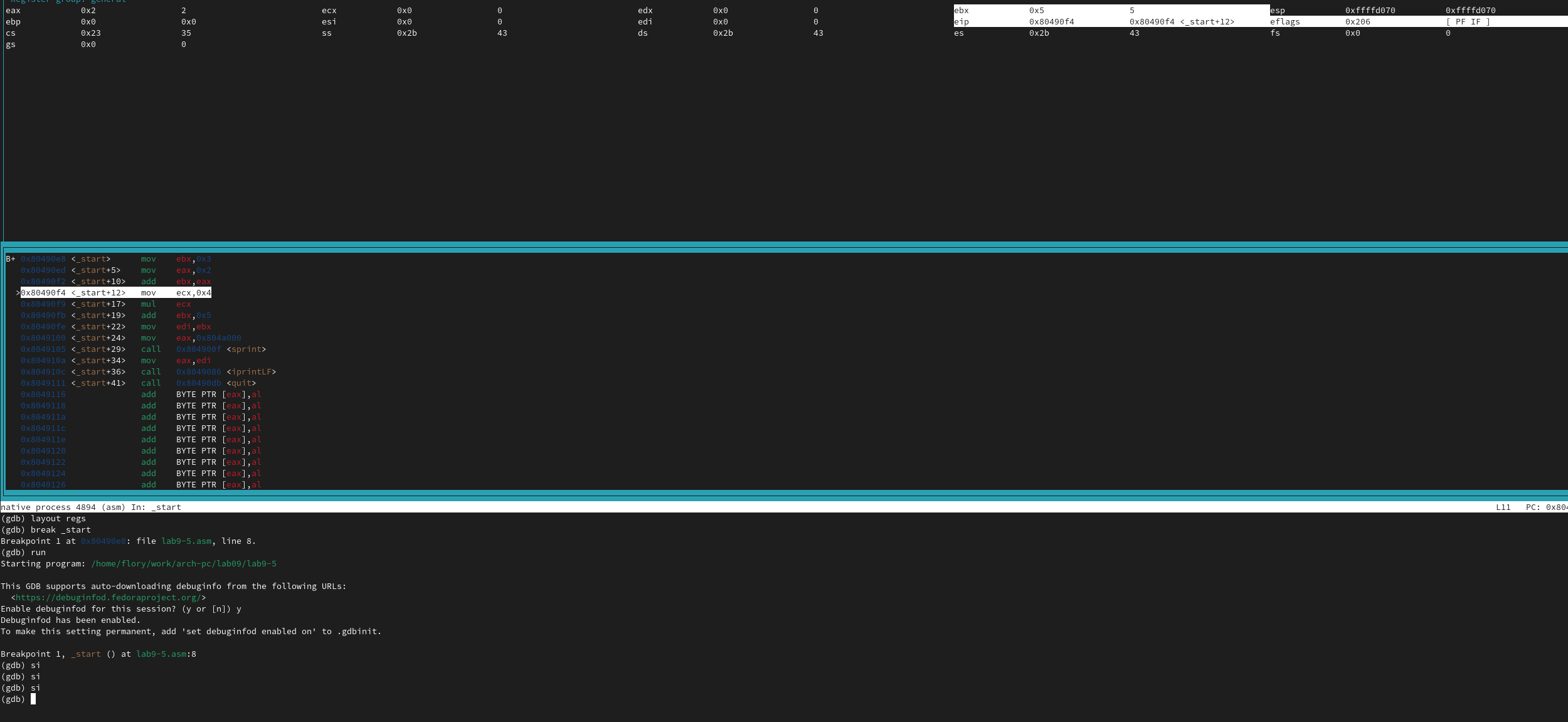
Значение1

1. Значение регистров на 2 шаге:(Рис. 3.8)



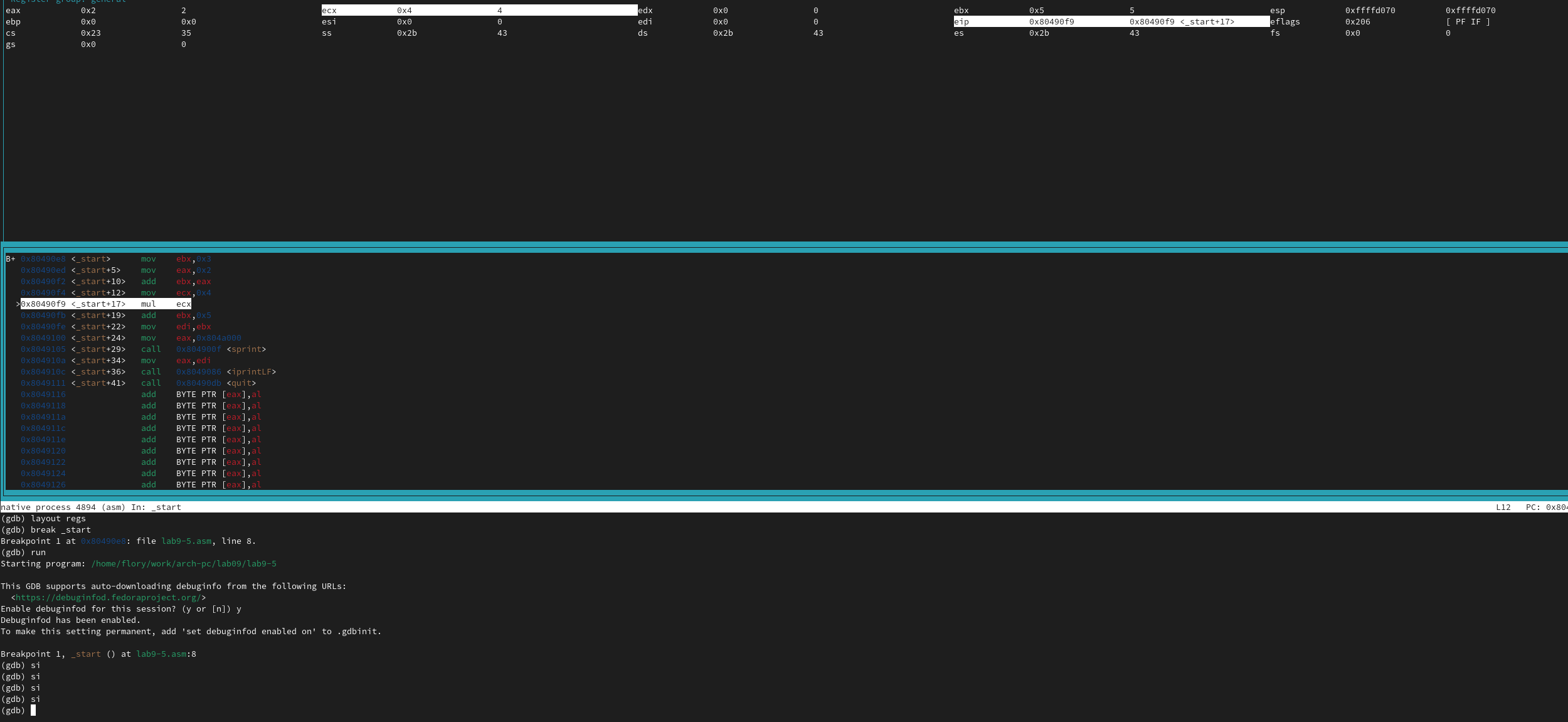
Значение2

1. Значение регистров на 3 шаге:(Рис. 3.9)



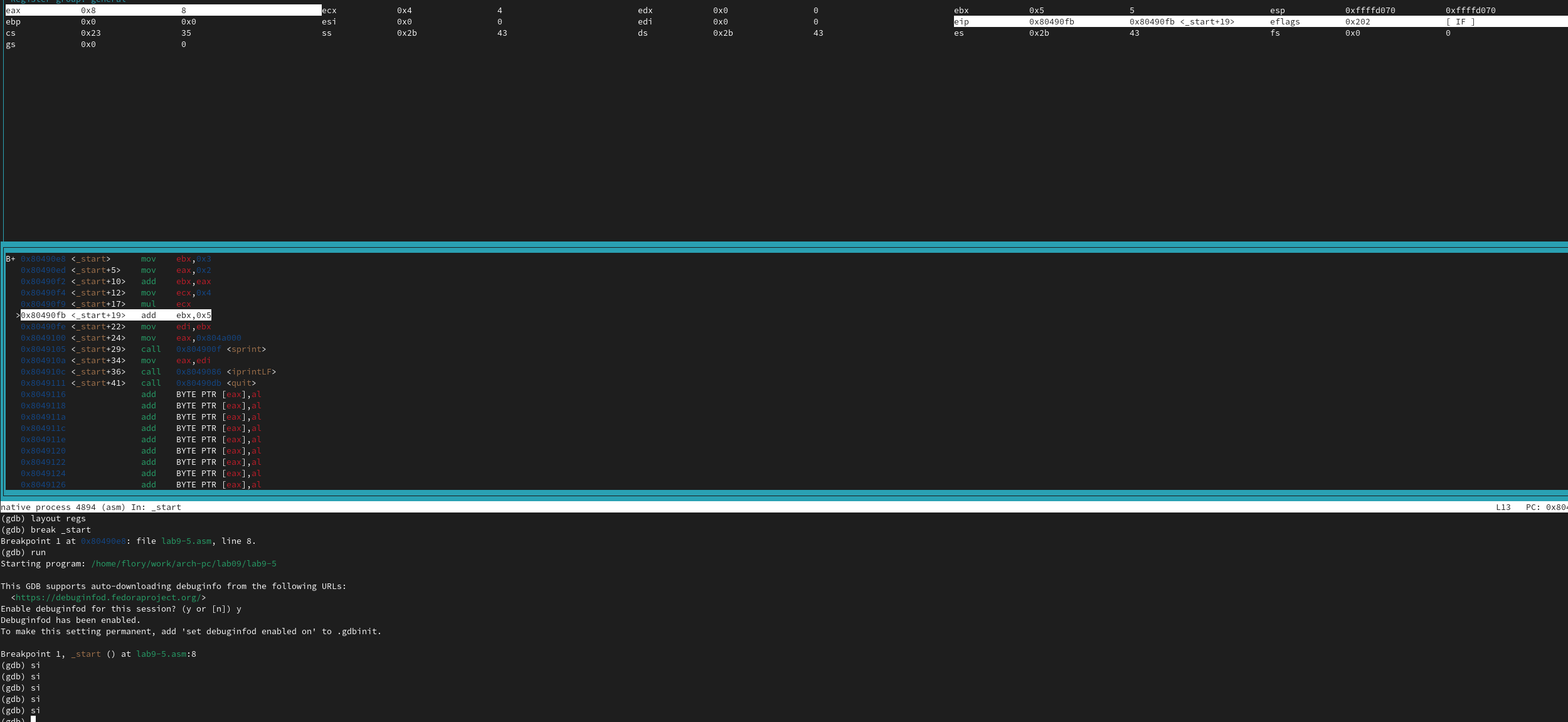
Значение3

1. Значение регистров на 4 шаге:(Рис. 3.10)



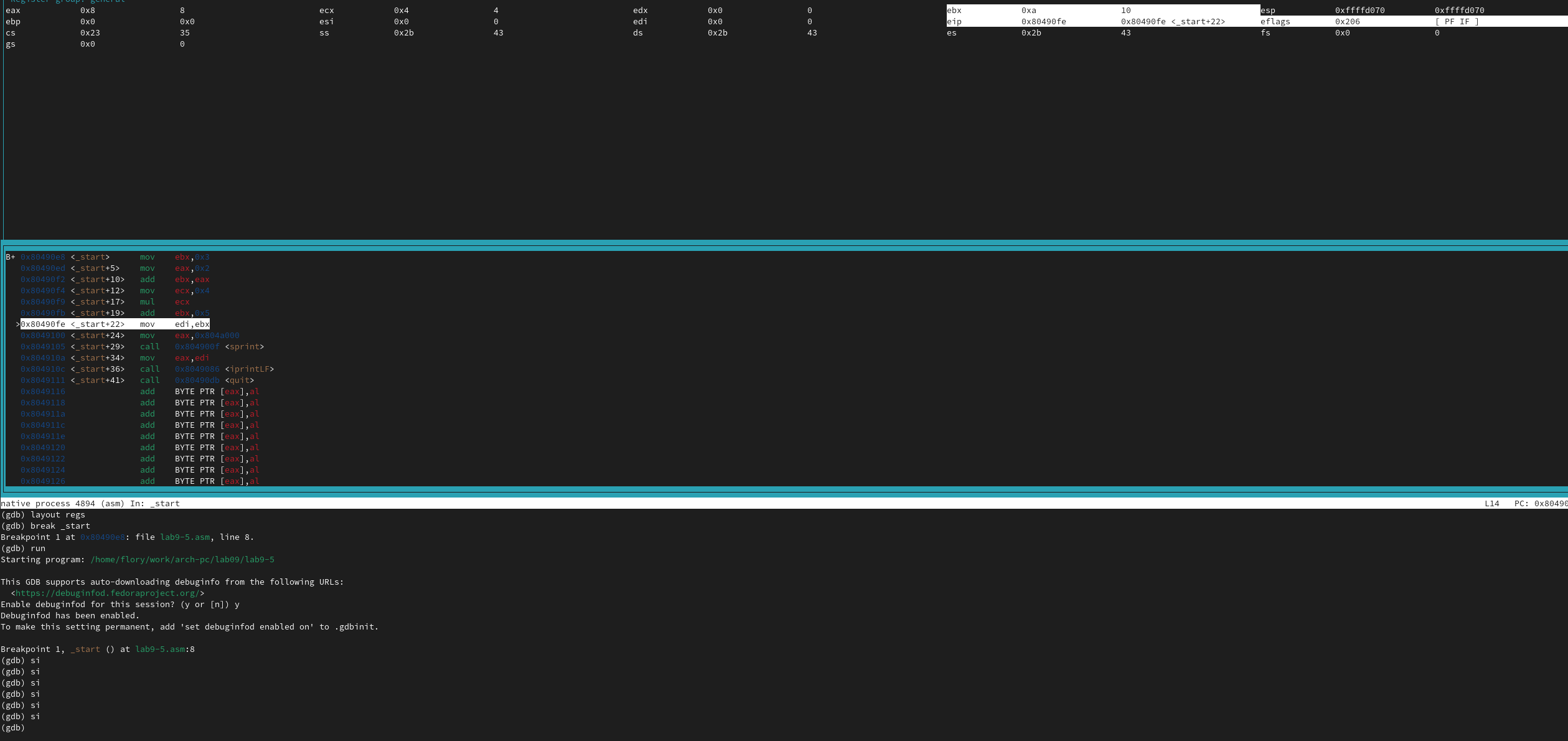
Значение4

1. Значение регистров на 5 шаге:(Рис. 3.11)



Значение5

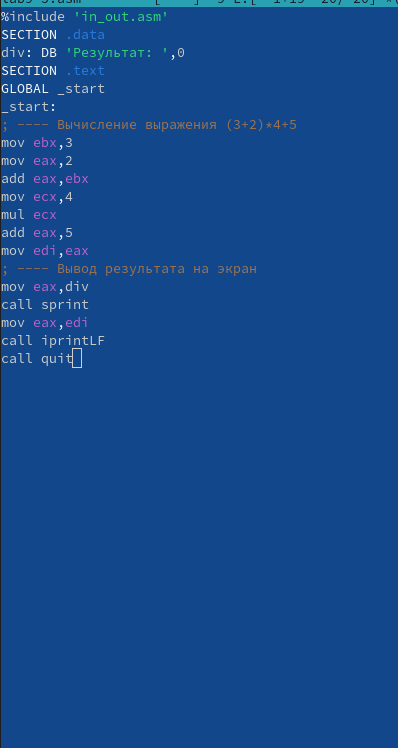
1. Значение регистров на 6 шаге:(Рис. 3.12)



Значение6

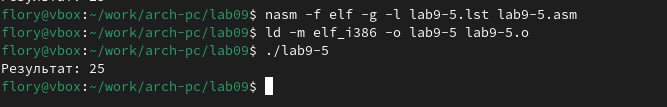
Ошибка заключается в том, что мы умножали значение регистра eax, а нужно было умножать значение регистра ebx. А результаты хранить в регистре eax.

1. Теперь изменю код:(Рис. 3.13)



Правильный код

1. Запущу его и вижу, что ответ правильный:(Рис. 3.14)



Правильный код

# 4 Вывод

В ходе выполнения лабораторной работы я приобрел навыки написания программ с использованием подпрограмм и ознакомился с методами отладки при помощи gdb и его основными возможностями.