Правительство Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
(НИУ ВШЭ)

Московский институт электроники и математики им. А.Н. Тихонова

ОТЧЕТ
О ПРАКТИЧЕСКОЙ РАБОТЕ № 2.1

по дисциплине «Архитектура компьютера и системное программирование»

ЗНАКОМСТВО С АССЕМБЛЕРОМ И РАЗНОЯЗЫКОВЫМИ ПРОГРАММАМИ

Студент гр. БИБ232
Николаев М.А

Руководитель
Абдуллаев Ариф Биннатович
«03» декабря 2025 г.

Москва 2025

СОДЕРЖАНИЕ

1 Задание на практическую работу

Цели настоящей практической работы заключаются в следующем
В рамках практической работы необходимо выполнить следующее:

1. Познакомиться со структурой программы на языке Ассемблер, разновидностями и назначением сегментов, способами организации простых и сложных типов данных, изучить форматы и правила работы с транслятором MASM / FASM /TASM, компоновщиком и отладчиком, познакомиться с средствами создания программ на Ассемблере для ОС Linux.

2. Исследование способов связывания разноязыковых модулей.
Задание настоящей практической работы заключаются в следующем:

1 Основы программирования на ассемблере:

1.1 Повторно ознакомиться с теоретическим материалом.

1.2 С помощью языка программирования Ассемблер, используя любой синтаксис (Intel или AT&T) на свой выбор, необходимо написать программу, решающую задачу из варианта. При этом:

1.2.1 Решением задачи является запись в специально объявленную для этого переменную верного результата при любых допустимых входных данных.

1.2.2 Программа должна корректно завершаться, не вызывая аварийный остановок.

1.2.3 Объявлять входные данные необходимо самостоятельно в секции данных. Например, если в задаче указано «дан массив из 10 байт», в секции данных необходимо самостоятельно объявить такой массив, заполнить его байтами на своё усмотрение и использовать в качестве входного. В ходе защиты может возникнуть необходимость изменения таких данных.

1.2.4 Для выполнения повторяющихся однотипных действий необходимо использовать соответствующие средства языка. Копирование и вставка одного и того же кода нужное количество раз с незначительными изменениями без необходимости является ошибкой.

1.2.5 Программа должна выдавать корректный ответ для всех видов входных данных, описанных в задании. Например, если в задании написано «дан массив из 10 байт», то входными данными может быть массив из любых 10 значений от 0 до 255. Учитывайте возможные переполнения!

1.2.6 При выполнении (в т.ч. дополнительных заданий) нельзя вызывать готовые ассемблерные подпрограммы из каких-либо библиотек и модулей.

2 Комбинирование разноязыковых модулей:

2.1 Повторно ознакомиться с теоретическим материалом.

2.2 С помощью языков программирования С и Ассемблер, используя любой синтаксис (Intel или AT&T) на свой выбор, необходимо написать программу, решающую задачу из варианта.

2.2.1 Входные данные для задач, основная часть которых подразумевает обработку чисел, а не их создание, должны генерироваться автоматически с помощью ГПСЧ языка Си. Сгенерированные исходные данные должны быть выведены в консоль.

2.2.2 При случайной генерации чисел пользователю должна предоставляться **возможность ограничить диапазон** значений этих чисел.

2.2.3 Решением задачи является вывод в консоль верного результата при любых допустимых входных данных.

2.2.4 Программа должна корректно завершаться, не вызывая аварийный остановок.

2.2.5 В программе на языке С должен быть реализован только следующий функционал: выделение памяти под исходные данные (и их инициализация в случаях, когда она должна быть произведена с помощью ГПСЧ, или ввод данных), вызов подпрограммы, выполняющей основные вычисления, вывод результата вычислений в консоль. Все основные вычисления должны быть оформлены в виде ассемблерного модуля (не ассемблерной вставки в код на Си, а именно отдельного файла на языке Assembler).

2.2.6 Для выполнения повторяющихся однотипных действий необходимо использовать соответствующие средства языка. Копирование и вставка одного и того же кода нужное количество раз с незначительными изменениями без необходимости является ошибкой.

2.2.7 Все программы должны работать с любыми размерами входных данных. Размер входных данных (даже если сами данные генерируются случайно)должен задаваться пользователем посредством ввода с командной строки.

3 Дополнительные задания:

3.1 Вывести результирующий элемент данных в консоль (в окно «вывод» в IDE), используя только команды «чистого» ассемблера (не вызывая подпрограммы,

написанные не самостоятельно, наподобие printf). Даёт два дополнительных балла.

3.2 Выполнить все поставленные задачи, написав две программы с использованием обоих синтаксисов (Intel и AT&T). Даёт один дополнительный балл.

3.3 Скомбинировать разноязыковые модули для ОС Linux и ОС Windows, используя любой компилятор, кроме GCC/G++ (а также иных средств, входящих в пакет GCC), и синтаксис Intel. При выполнении данного задания необходимо подробно пошагово отразить все этапы в отчёте. Даёт два дополнительных балла.

2 Ход работы

2.1 Основы программирования на ассемблере

Мой порядковый номер в группе: 51.

Задание: «Напишите программу, в которой создается двумерный числовой массив и для этого массива вычисляется сумма квадратов его элементов.»

2.1.1 Алгоритм

1. Инициализировать переменную для хранения суммы остатков.
2. Внутри цикла посчитать все остатки, добавляя их к ранее инициализированной переменной
3. Вывести итог в консоль.

2.1.2 Процесс разработки

1. Разработка на ОС Linux, для Linux.
2. Написание исходного кода (приложен к ответу на практическую работу) синтаксисом Intel
3. Сборка средствами fasm и ld.
4. Написание Makefile для автоматизации сборки (приложен к ответу на практическую работу).

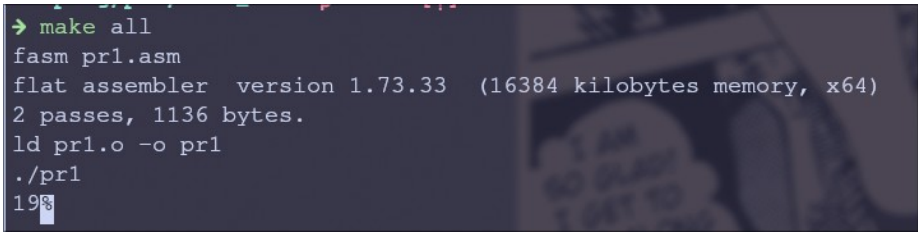

Рисунок 1 — Сборка и запуск первого задания при помощи Makefile

2.2 Комбинирование разноязыковых модулей

Мой порядковый номер в группе: 51.

Задание: «Напишите программу, в которой создается квадратная матрица (реализуется через двумерный массив). Матрица заполняется случайными числами, после чего выполняется «поворот по часовой стрелке»: инвертированный первый столбец становится первой строкой, инвертированный второй столбец становится второй строкой, и так далее.»

2.2.1 Алгоритм

1. Часть C:
   1.1. Запросить у пользователя размер матрицы.
   1.2. Запросить у пользователя минимальное и максимальное ограничение для случайного числа.
   1.3. Выделить память для матрицы.
   1.4. Сгенерировать матрицу.
   1.5. Вызывать assembler функцию, выполняющую поворот матрицы
   1.6. Вывести результат в консоль
   1.7. Освободить выделенную память
2. Часть asm:
   2.1. Выделить часть стека
   2.2. Последовательно переложить значения из одной матрицы во вторую
   2.3. Передать результат в часть C

2.2.2 Процесс разработки

1. Разработка на ОС Linux, для Linux.
2. Написание исходного кода (приложен к ответу на практическую работу) синтаксисом Intel
3. Сборка средствами fasm и gcc.
4. Написание Makefile для автоматизации сборки (приложен к ответу на практическую работу).

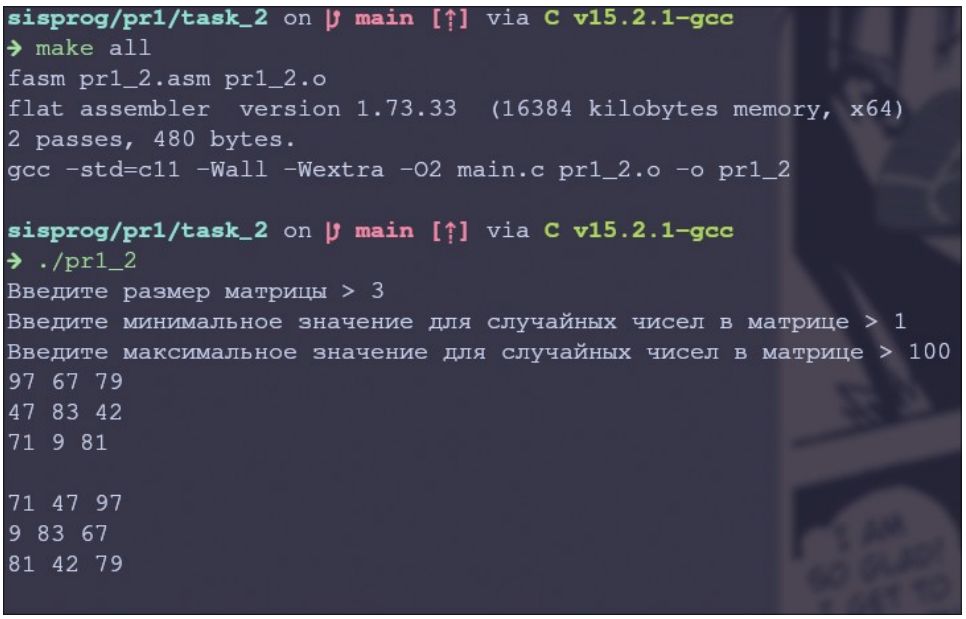

Рисунок 2 — Сборка и запуск второго задания при помощи Makefile

3. Выводы о проделанной работе

В ходе практической работы были изучены основы программирования на ассемблере и реализованы алгоритмы обработки данных с использованием низкоуровневых инструкций. Также было освоено взаимодействие модулей на C и ассемблере, что позволило понять принципы комбинирования разноязыковых программ