Лукьянчук Богдан БПИ-213

Отчёт

Идз №2

Вариант № 12

4 балла:

1. Решение задачи на C представлено в файле main.c.
2. С помощью флага gcc -S -O0 -masm=intel -fno-asynchronous-unwind-tables -fcf-protection=none компилирую Assembly код из C кода. В папке main_without_optimize файле main_without_Optimize.s находится сам Assembly код с поясняющими комментариями, так же в папке располагаются перемещаемый объектный файл main_without_Optimize.o и исполняемый объектный файл main_without_Optimize.
3. При использовании флага из 2 пункта были убраны лишние макросы.
4. Модифицированная ассемблерная программа находится в папке main_without_optimize, описание которой представлено во 2 пункте.
5. Для тестирования программы была создана папка Tests с input.txt (готовые входные данные) и output.txt (правильные выходные данные, соответствующие входным) файлами. С помощью скрипта на bash, который представлен в файле test.sh, производилась проверка на корректность работы программ, программа получала на вход данные из input.txt и записывала выходные данные в файл out.txt, который позже сравнивался с output.txt. Наглядно можно увидеть корректность работы программ, используя команды Makefile: make testC (тестирует программу main.c), make testAsm (тестирует програму main_without_Optimize.s).

5 баллов

1. Изначально программа была разбита на функции readStr, readStrFile, findMinMax, generateStr, writeStr, writeStrFile, main. Из функции main по очереди вызывались данные функции, в которые передавались, соответствующие данные (полное описание принимаемых аргументов функции, вызов функций и передачу аргументов им можно увидеть в файле main.c).
2. В функции findMinMax были инициализированы локальные переменные charMax, sizeStr, i. В функции generateStr инициализирована локальная перемення i.

3. В файле main_wthout_Optimize.s, который располагается в папке main_with_registers, описаны все передачи фактических параметров и перенос возвращаемого результата, а также добавлены комментарии, описывающие связь между параметрами языка С и регистрами(стеком).

6 баллов:

1. С помощью сохраняемых регистров в функциях: readStr, readStrFile, findMinMax, generateStr, writeStr, writeStrFile, все используемые переменные перемещались в эти регистры, а не на стек. Полное описание изменения ассемблерного кода за счет использования регистров можно увидеть в файле main_with_Registers.s, который располагается в папке main_with_registers. Также в папке представлены исполняемый файл и перемещаемый файл данной программы.
2. Для тестирования программы была создана папка Tests с input.txt (готовые входные данные) и output.txt (правильные выходные данные, соответствующие входным) файлами. С помощью скрипта на bash, который представлен в файле test.sh, производилась проверка на корректность работы программ, программа получала на вход данные из input.txt и записывала выходные данные в файл out.txt, который позже сравнивался с output.txt. Наглядно можно увидеть корректность работы программы, используя команду Makefile: make testAsmReg.

7 баллов:

1. Свою программу модифицирую для того, чтобы она могла считывать команды и расположение файлов из командной строки. Программа считывает следующие аргументы из командной строки: **1)** file input.txt(расположение файла с входными данными) output.txt(расположение файла, куда необходимо записать результат работы); **2)** без аргументов (программа ожидает входные данные со стандартного потока ввода stdin и выводит результат работы в stdout). Для примера работы программы в Makefile создана инструкция ReadWriteFile, которая запускает программу и передает ей аргументы **1)** file InOutputFiles/input.txt InOutputFiles/output.txt. Здесь представлена передача расположения файлов, откуда мы берем данные, и куда мы записываем результат. Результат выполнения инструкции make ReadWriteFile можно увидеть в файлах output.txt и output_sec.txt, которые расположены в папке InOutputFiles.
2. Данная программа была разбита на две единицы компиляции: main_part, second_part, в первой располагается функция main с вызовами определенных

функций, во второй единице компиляции располагаются все вызываемые функцией main функции. Код обеих единиц компиляции, а также слинкованный исполняемый файл complete представлены в папке main_with_parts.s.

8 баллов:

1. В исходную программу была добавлена функция generateStr, принимающая адрес строки, которую необходимо заполнить, а также размер массива, который нужно сгенерировать (размер не должен превышать 1024*1024 – 1). Генератор подключается при появлении команды random в командной строке, результат записывается в stdout.
2. Командная строка имеет несколько ключей, с которыми работает: **1)** file input.txt(расположение файла с входными данными) output.txt(расположение файла, куда необходимо записать результат работы); **2)** random size (программа генерирует строку размера size случайным образом, результат выводится в stdout); **3)** без аргументов (программа ожидает входные данные со стандартного потока ввода stdin и выводит результат работы в stdout.).
3. В исходные программы была добавлена модификация, которая производит замер времени работы функции findMinMax. При запуске программы с первым ключом time программа будет работать в режиме замера времени, все остальные ключи также работают вместе с time, если вводить их следующими аргументами при запуске. Для замера производительности разных вариантов программы, время работы основной (не модифицированной) программы main.c было увеличено до ~ 1 секунды с помощью зацикливания выполнения функции findMinMax и специально подобранных размеров строк. Всего сравнивалось три модификации программы: main (не модифицированная версия), main_with_Registers(версия с максимальным использованием регистров процессора), complete (версия, разбитая на единицы компиляции). По результатам замеров времени все программы запускались с ключом random для генерации случайных массивов размера: 550, 500, 400 элементов. Main выполняется ~ 1.3 – 1 секунды, main_with_Registers за 0.9 – 0.65 секунды, complete ~ за 0.9 – 0.65. Для предоставления результатов замеров работы программ в Makefile прописана инструкция specialDatatime, которая запускает эти программы с опцией time и случайными массивами размера: 550, 500, 400 элементов.