Отчет по индивидуальному домашнему заданию №1 по "Архитектуре вычислительных систем"

Владимиров Дмитрий Андреевич, БПИ218 ВАРИАНТ 14

Условие. Сформировать массив B из элементов массива A заменой всех отрицательных значений на максимум из массива A.

На 4 балла:

1. Написан код решения задачи. Исходный код программы на языке C расположен в файле main.c

2. Файл main.c откомпилирован в ассемблерную программу без каких-либо оптимизаторов путем команды > gcc -O0 -Wall -masm=intel -S main.c -o main.s. Результат лежит в файле main.s, в нем в комментариях указана связь переменных в коде на C с действиями с ними в коде на ассемблере.

3. Файл main.c скомпилирован в исполняемый файл main.out путем команды > gcc -O0 -Wall main.c -o main.out.

4. Также файл main.c скомпилирован в ассемблерную программу с учетом оптимизации путем команды > gcc -masm=intel -fno-asynchronous-unwind-tables -fno-jump-tables -fno-stack-protector -fno-exceptions main.c -S -o main_opt.s. Файл main_opt.s тоже прикреплен.

5. Далее программу main.c скомпилирован в исполняемый файл main_opt.out путем команды > gcc -masm=intel -fno-asynchronous-unwind-tables -fno-jump-tables -fno-stack-protector -fno-exceptions main.c -o main_opt.out.

6. Остается только протестировать оба исполняемых файла. оба файла main.out и main_opt.out прикреплены, поэтому можно убедиться в правильности приведенных данных.

Входные данные	main.out	main_opt.out
1 2 3 4 5	1 2 3 4 5	1 2 3 4 5
	not enough space or empty array	not enough space or empty array
5 4 3 2 -9	5 4 3 2 5	5 4 3 2 5

-1 2 3 0 -2	3 2 3 0 3	3 2 3 0 3
1 -1 -1 -1 -1 -1 -1	1 -1 -1 -1 -1 -1 -1	1 -1 -1 -1 -1 -1 -1
-5 -6 -3 -2 -7	-5 -6 -3 -2 -7	-5 -6 -3 -2 -7
8 8 9 -1 -1000000 4 5 6	8 8 9 9 9 4 5 6	8 8 9 9 9 4 5 6
-3 4 2 8 9	9 4 2 8 9	9 4 2 8 9

Как видим, программы работают идентично

На 5 баллов:

1. Изменил текст программы main.c, добавив туда функции print_array и fill. Теперь будем работать с программой main2.c.
2. Локальные переменные использовались еще в предыдущей программе, в этой так же присутствуют.
3. Скомпилировал в ассемблерную программу с оптимизацией при помощи команды > gcc -masm=intel -fno-asynchronous-unwind-tables -fno-jump-tables -fno-stack-protector -fno-exceptions main2.c -S -o main2.s.
4. Также в файле main2.s дописал комментарии, в каких местах передаются параметры в функцию и как хранятся там (указал регистры).

На 6 баллов:

Задача: использовать регистры процессора везде, где можно, то есть минимизировать использование стека.

1. Где было возможно сохранил переменные в регистры процессора, в особенности r12, r13 и rbx. Файл с ассемблерным кодом после рефакторинга и комментариями – main2_refactored.s.

2. Протестируем программу на тех же входных данных, что и в первом случае (когда тестировали программу на 4 балла), предварительно скомпилировав ее при помощи команды > gcc main2_refactored.s -o main2_refactored.out.

Входные данные	Результат программы
1 2 3 4 5	1 2 3 4 5
	not enough space or empty array

5 4 3 2 -9	5 4 3 2 5
-1 2 3 0 -2	3 2 3 0 3
1 -1 -1 -1 -1 -1 -1	1 -1 -1 -1 -1 -1 -1
-5 -6 -3 -2 -7	-5 -6 -3 -2 -7
8 8 9 -1 -1000000 4 5 6	8 8 9 9 9 4 5 6
-3 4 2 8 9	9 4 2 8 9

Итак, результаты полностью совпадают с первыми, а значит программа работает корректно.