МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра ИИТ

Отчёт

о лабораторной работе №1

по дисциплине «Компьютерные системы и сети»

Тема: «Изучение аппаратно-программной архитектуры процессоров семейства Intel – разработка ассемблерной вставки»

Выполнил студент 2 курса

группы ПО-11Сымоник И.А.

Номер зачетной книжки: 220220

Проверил: Савицкий Ю.В.

**Цель работы**: изучение программной модели микропроцессора, регистров, организации памяти; практическая работа с ассемблерными вставками

**Вариант 6**

**Ход работы**

**Задание:** Нечетные символы заменить на символ +.

**Исходный код:**

#include <iostream>

#include <string>

#include <chrono>

#include <Windows.h>

#include <string\_view>

void AssemblyFunc(const char inputString[])

{

std::chrono::steady\_clock::time\_point begin = std::chrono::steady\_clock::now();

\_\_asm

{

mov esi, inputString

mov ecx, 0

replace:

mov al, [esi + ecx]

test al, al

jz end\_replace

test ecx, 1

jnz replace\_char

inc ecx

jmp replace

replace\_char :

mov[esi + ecx], '+'

inc ecx

jmp replace

end\_replace :

}

std::chrono::steady\_clock::time\_point end = std::chrono::steady\_clock::now();

std::cout << "Время работы алгоритма : "

<< std::chrono::duration\_cast<std::chrono::seconds>(end - begin).count()

<< " секунд "

<< std::chrono::duration\_cast<std::chrono::milliseconds>(end - begin).count()

<< " миллисекунд "

<< std::chrono::duration\_cast<std::chrono::microseconds>(end - begin).count()

<< " микросекунд "

<< std::chrono::duration\_cast<std::chrono::nanoseconds>(end - begin).count()

<< " наносекунд "

<< std::endl;

std::cout << "Измененная строка: " << inputString << std::endl;

}

void Func(std::string& str)

{

std::chrono::steady\_clock::time\_point begin = std::chrono::steady\_clock::now();

auto size = str.size();

for (int i = 1; i < size; i+=2)

{

str[i] = '+';

}

std::chrono::steady\_clock::time\_point end = std::chrono::steady\_clock::now();

std::cout << "Время работы алгоритма : "

<< std::chrono::duration\_cast<std::chrono::seconds>(end - begin).count()

<< " секунд "

<< std::chrono::duration\_cast<std::chrono::milliseconds>(end - begin).count()

<< " миллисекунд "

<< std::chrono::duration\_cast<std::chrono::microseconds>(end - begin).count()

<< " микросекунд "

<< std::chrono::duration\_cast<std::chrono::nanoseconds>(end - begin).count()

<< " наносекунд "

<< std::endl;

std::cout << "Измененная строка: " << str << std::endl;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

std::string\_view str = "dsakdklas;kd;sajdlkjsadhjlsahdjkhsajkdhksjahdkjhaskhdjkhdkjhasdkjhasjkdhjkashdkhasdhjkashdjkassajdlkjsadhjlsjkdhksjahdkjhasjkdhjkashdkhasdhjkashdjkasdsakdklas;kd;sajdlkjsadhjlsahdjkhsajkdhksjahdkjhaskhdjkhdkjhasjkdhksjahdkjhasjkdhjkashdkhasdhjkashdjkassajdlkjsadhjlsahdjkhsajkdhksjahdkjhaskhdjkhdkjhasjkdhksjahahdjkhsajkdhksjahdkjhaskhdjkhdkjhasjkdhksjahdkjhasjkdhjkashdkhasdhjkashdjkasdsadsadddawhc";

std::string inputString = { str.data() };

std::cout << "Оригинальная строка: " << inputString << std::endl;

std::cout << std::endl;

std::cout << "Функция с ассемблерной вставкой: " << std::endl;

AssemblyFunc(inputString.c\_str());

inputString = { str.data() };

std::cout << std::endl;

std::cout << "Функция без ассемблерной вставки: " << std::endl;

Func(inputString);

return 0;

}

**Результат выполнения:**

Изображение выглядит как текст, снимок экрана, Шрифт, черно-белый

Автоматически созданное описание

Скорость выполнения функции без ассемблерной вставки выше. Это можно объяснить тем, что к ассемблерным вставкам не применяются оптимизации компилятора.

**Вывод:** изучили программную модель микропроцессора, регистров, организации памяти.