федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет «Московский институт электронной техники»

Направление подготовки 09.03.04 «Программная инженерия»

Дисциплина «Электротехника»

**Лабораторная РАБОТА №5**

**Использование ассемблерных вставок в программах на C++. Передача параметров во вставку. Перезаписываемые элементы**

Работу выполнил студенты группы ПИН-24 Баранов Д.А. и Демочкина А.В.

Работу проверил ассистент Института СПИНТех Фомин Р.А.

**Цель работы:** научиться вставлять в программы на языке высокого уровня ассемблерные фрагменты.

Для выполнения заданий выбран онлайн компилятор <https://godbolt.org/>.

ОС и разрядность ОС: GNU/Linux 64

Компилятор: Compiler Explorer x86-64 gcc 11.2

Архитектура: x86-64

**Задание 1.**

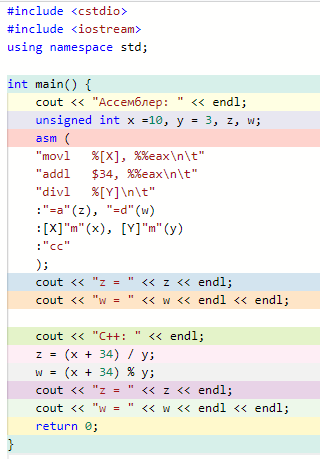
**Вариант 2**

****

Реализуйте расчёт беззнакового целочисленного выражения как ассемблерную вставку в программу на C/C++. При этом 𝑥, 𝑦, 𝑧, 𝑤 — 32-битные локальные переменные функции 𝑚𝑎𝑖𝑛() (или другой функции C/C++) и передаются во вставку как параметры (𝑧 и 𝑤 — выходные, 𝑥 и 𝑦 — входные).

Входные параметры, соответствующие 𝑥 и 𝑦 (дайте им псевдонимы [X] и [Y]), должны располагаться в памяти. Затем, внутри вставки, при необходимости скопируйте [X] и [Y] в регистры общего назначения.

Проверьте расчёт, реализовав то же самое на С/C++.



#include <cstdio>

#include <iostream>

using namespace std;

int main() {

cout << "Ассемблер: " << endl;

unsigned int x =10, y = 3, z, w;

asm (

"movl %[X], %%eax\n\t"

"addl $34, %%eax\n\t"

"divl %[Y]\n\t"

:"=a"(z), "=d"(w)

:[X]"m"(x), [Y]"m"(y)

:"cc"

);

cout << "z = " << z << endl;

cout << "w = " << w << endl << endl;

cout << "C++: " << endl;

z = (x + 34) / y;

w = (x + 34) % y;

cout << "z = " << z << endl;

cout << "w = " << w << endl << endl;

return 0;

}

Ассемблер:

z = 14

w = 2

C++:

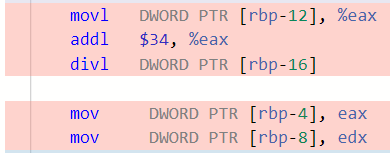
z = 14

w = 2

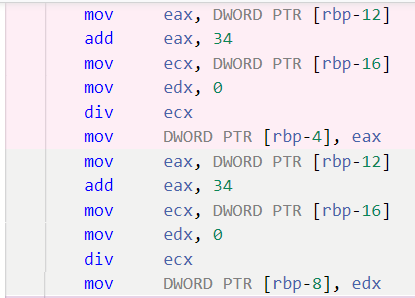
Помещение значений переменных x, y в память:



Отображение ассемблеровской вставки:



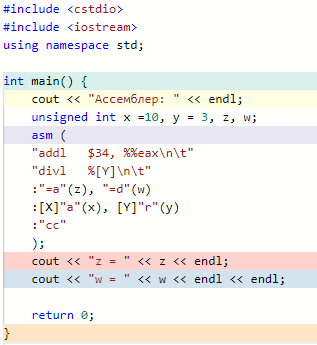
Отображение программы, реализованной на C++:



Обращение к параметрам в памяти [X] и [Y]: %[X], %[Y]

**Задание 2.**

Реализуйте задание Л5.з1, располагая параметры [X] и [Y] в регистрах общего назначения ([X] в регистре 𝐴, [Y] — в выбираемом компилятором)

****

**#include <cstdio>**

**#include <iostream>**

**using namespace std;**

**int main() {**

**cout << "Ассемблер: " << endl;**

**unsigned int x =10, y = 3, z, w;**

**asm (**

**"addl $34, %%eax\n\t"**

**"divl %[Y]\n\t"**

**:"=a"(z), "=d"(w)**

**:[X]"a"(x), [Y]"r"(y)**

**:"cc"**

**);**

**cout << "z = " << z << endl;**

**cout << "w = " << w << endl << endl;**

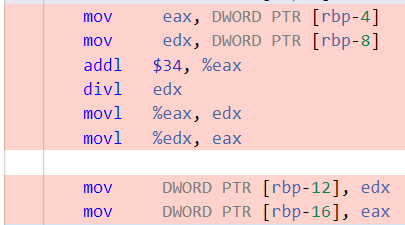
**return 0;**

**}**

Ассемблер:

z = 14

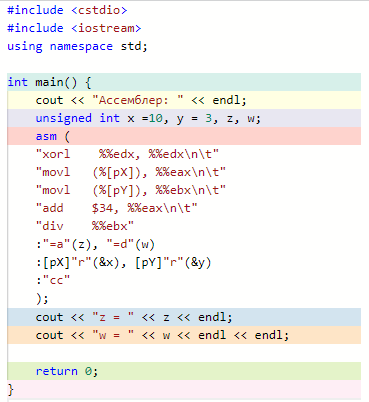
w = 2

****

Параметр [X] расположен в регистре общего назначения eax. Параметр [Y] расположен в регистре, выбираемом компилятором (в регистре edx).

Обращение к параметрам [X] и [Y]: %[X], %[Y]

**Задание 3.**

****

**#include <cstdio>**

**#include <iostream>**

**using namespace std;**

**int main() {**

**cout << "Ассемблер: " << endl;**

**unsigned int x =10, y = 3, z, w;**

**asm (**

**"xorl %%edx, %%edx\n\t"**

**"movl (%[pX]), %%eax\n\t"**

**"movl (%[pY]), %%ebx\n\t"**

**"add $34, %%eax\n\t"**

**"div %%ebx"**

**:"=a"(z), "=d"(w)**

**:[pX]"r"(&x), [pY]"r"(&y)**

**:"cc"**

**);**

**cout << "z = " << z << endl;**

**cout << "w = " << w << endl << endl;**

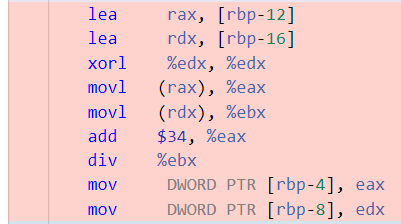
**return 0;**

**}**

Ассемблер:

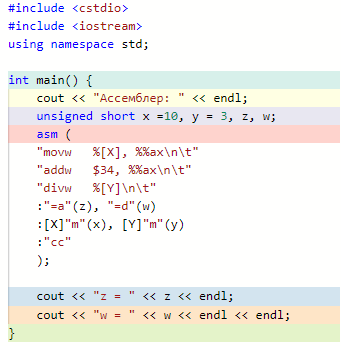
z = 14

w = 2

****

**Задание 4.**

Реализуйте задание Л5.з1 для 16-битных 𝑥, 𝑦, 𝑧, w.



#include <cstdio>

#include <iostream>

using namespace std;

int main() {

cout << "Ассемблер: " << endl;

unsigned short x =10, y = 3, z, w;

asm (

"movw %[X], %%ax\n\t"

"addw $34, %%ax\n\t"

"divw %[Y]\n\t"

:"=a"(z), "=d"(w)

:[X]"m"(x), [Y]"m"(y)

:"cc"

);

cout << "z = " << z << endl;

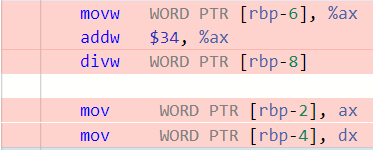
cout << "w = " << w << endl << endl;

}

Ассемблер:

z = 14

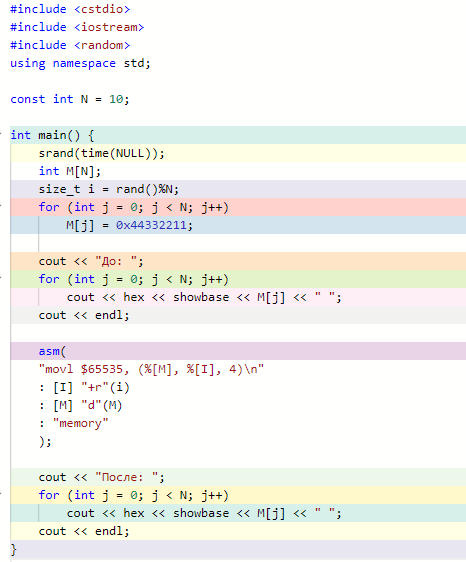
w = 2



**Задание 5.**

На языке C/C++ выделите память под массив 𝑀 (статический или динамический) из 𝑁 32-битных целых чисел и инициализируйте 𝑀 значениями 0x44332211.

Реализуйте для заданного 𝑘 ∈ [0, 𝑁) запись значения 𝑥 ̸= 0 на место элемента 𝑀[𝑘], используя компоненты эффективного адреса (𝐵𝑎𝑠𝑒, 𝐼𝑛𝑑𝑒𝑥, 2 𝑆𝑐𝑎𝑙𝑒).

****

**#include <cstdio>**

**#include <iostream>**

**#include <random>**

**using namespace std;**

**const int N = 10;**

**int main() {**

**srand(time(NULL));**

**int M[N];**

**size\_t i = rand()%N;**

**for (int j = 0; j < N; j++)**

**M[j] = 0x44332211;**

**cout << "До: ";**

**for (int j = 0; j < N; j++)**

**cout << hex << showbase << M[j] << " ";**

**cout << endl;**

**asm(**

**"movl $65535, (%[M], %[I], 4)\n"**

**: [I] "+r"(i)**

**: [M] "d"(M)**

**: "memory"**

**);**

**cout << "После: ";**

**for (int j = 0; j < N; j++)**

**cout << hex << showbase << M[j] << " ";**

**cout << endl;**

**}**

**1 запуск:**

****

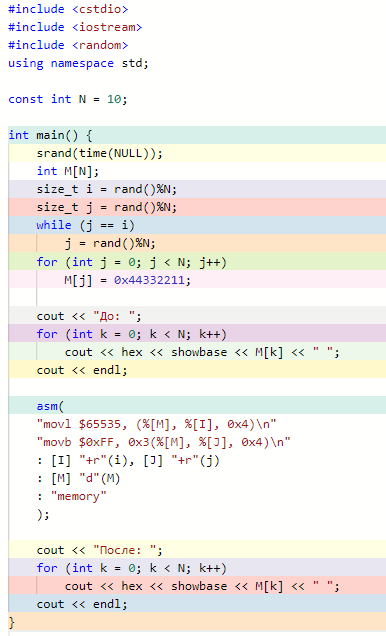
**2 запуск:**  


**3 запуск:**

****

**Задание 6.**

Реализуйте для заданного 𝑗 ∈ [0, 𝑁), 𝑗 ̸= 𝑘 запись значения FF в старший байт элемента 𝑀[𝑗], используя все компоненты эффективного адреса.



#include <cstdio>

#include <iostream>

#include <random>

using namespace std;

const int N = 10;

int main() {

srand(time(NULL));

int M[N];

size\_t i = rand()%N;

size\_t j = rand()%N;

while (j == i)

j = rand()%N;

for (int j = 0; j < N; j++)

M[j] = 0x44332211;

cout << "До: ";

for (int k = 0; k < N; k++)

cout << hex << showbase << M[k] << " ";

cout << endl;

asm(

"movl $65535, (%[M], %[I], 0x4)\n"

"movb $0xFF, 0x-1(%[M], %[J], 0x4)\n"

: [I] "+r"(i), [J] "+r"(j)

: [M] "d"(M)

: "memory"

);

cout << "После: ";

for (int k = 0; k < N; k++)

cout << hex << showbase << M[k] << " ";

cout << endl;

}

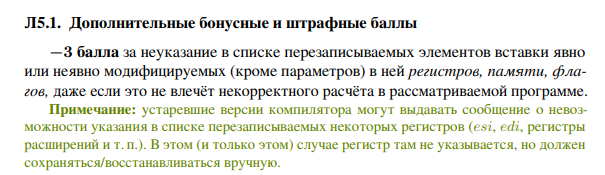
**1 запуск:**

****

**2 запуск:**  


**3 запуск:**

****

****

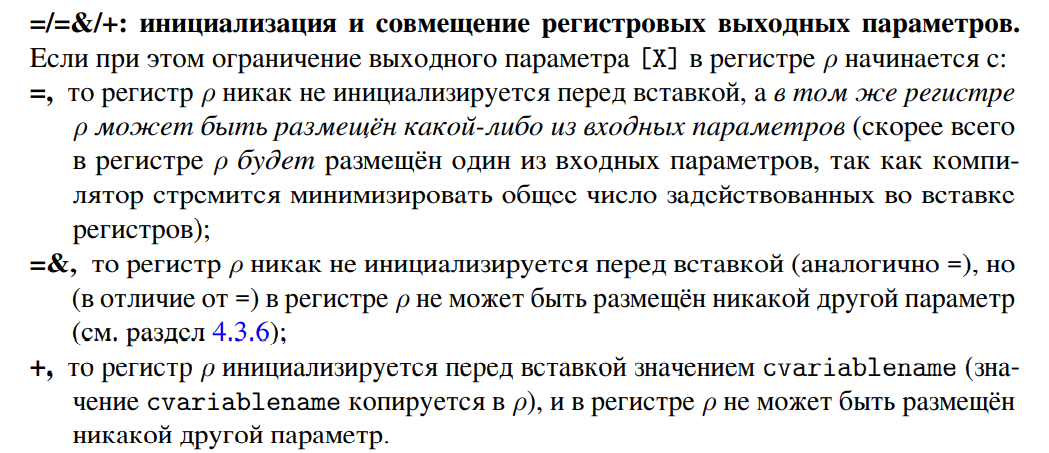
**Вопросы.**

1. Каким ключевым словом открывается ассемблерная вставка?

asm либо \_\_asm\_\_

2. Где описываются выходные параметры ассемблерных вставок расширенного синтаксиса GCC? Что означают символы =, =&, + в начале строки ограничений выходного параметра?

\_\_asm\_\_ (вставка : список\_выходных\_операндов : список\_входных\_операндов : список\_разрушаемых\_регистров );



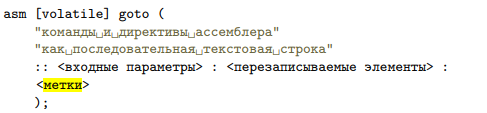
3. Где описываются входные параметры?

\_\_asm\_\_ (вставка : список\_выходных\_операндов : список\_входных\_операндов : список\_разрушаемых\_регистров );

4. Где указывается список перезаписываемых [clobbers] во вставке регистров (кроме параметров)? Какая строка соответствуют изменению флагов 𝑓𝑙𝑎𝑔𝑠? Какая строка соответствуют изменению памяти (кроме параметров)?

\_\_asm\_\_ (вставка : список\_выходных\_операндов : список\_входных\_операндов : список\_разрушаемых\_регистров );

5. Где описываются метки ЯВУ, на которые может быть передано управление из вставки?



6. Какое ключевое слово нужно указать после asm, чтобы запретить компилятору оптимизировать вставку?

asm [volatile] (...