**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМ**

**ЛАБОРАТОРНА РОБОТА №1.1.**

**Тема: Програмування лінійних процесів на Асемблері**

Виконав:

Студент групи 3СТ Чума Д. І.

Перевірив:

Асист. Ваш Ю. В.

Ужгород 2022

**Мета роботи:** Ознайомити студентів з основними командами мови Асемблер та навчитися програмувати лінійний процес.

1. **Теоретичні відомості.**

При організації лінійних процесів на мові Асемблер використовують такі команди, як: mov – команда передачі значення; add – команда додавання; sub – команда віднімання; mul – команда множення; div – команда ділення; xchg – команда обміну значень.

**1. Команди передачі значень:**

1) команда mov (передача): mov op1, op2.

По команді mov на місце першого операнда передається значення іншого операнда: op1:=op2. Команда не впливає на прапорці.

Приклади:

mov AX,30; A:=30

mov BL,DH; BL:=DH

По команді mov можна передавати як байт, так і слово.

2) команда xchg (обмін значень): xchg op1,op2.

Ця команда обмінює значення своїх операндів (вони повинні бути байтами, або словами);

op1 <=> op2. Команда не впливає на прапорці.

Приклади:

mov AX,62; AX=62

mov SI,135; SI=135

xchg AX,SI; AX=135, SI=62

**2. Команди додавання і віднімання.**

Команда add (додавання): add op1,op2

Команда sub (віднімання): sub op1,op2

Команда add додає операнди та заносить їхню суму на місце першого операнда: op1:=op1+op2.

По команді sub від першого операнда віднімається другий операнд, здобута різниця заноситися на місце першого операнда:

op1:=op1-op2.

Приклад:

add AH,12; AH:=AH+12

sub SI,z; SI:=SI-z

add z,-300; z:=z+(-300)

Команди add та sub працюють як з числами розміром у байт, так і з числами розміром у слово. Неможна, щоб один з операндів був байтом, а інший - словом.

**3. Команди множення й ділення.**

1) команда mul (множення): mul op

Операнд op, зазначений у команді - це лише один зі співмножників; він може знаходитись у регістрі чи в пам'яті, але не може бути безпосереднім операндом. Місцезнаходження іншого співмножника зафіксовано, і тому в команді не вказується. При множенні цілих операндів зі знаком, використовується команда imul op.

2) команда div (ділення): div op

Аналогічно команді множення.

1. **Текст програми**

Програма розроблена на мові програмування C++.

Варіант 17: f = (11y + 7x – 2z) / (y + 1).

#include <iostream>

using namespace std;

int main()

{

int x, y, z, f;

std::cout << "Input x=, y=, z=\n";

std::cin >> x >> y >> z;

//17. f = (11y + 7x - 2z) / (y + 1);

\_asm {

mov eax, 11 // ax = 11

imul y // ax = 11y

xchg eax, ebx // змінили ах і bx, bx = 11y

mov eax, 7 // ax = 7

imul x // ax = 7x

xchg eax, ebx // ax = 11y, bx = 7x

add eax, ebx // ax = 11y + 7x

mov ebx, 2 // bx = 2

xchg eax, ebx // ax = 2, bx = 11y + 7x

imul z // ax = 2z

sub ebx, eax // bx = 11y + 7x - 2z

mov ecx, y // cs = y

mov eax, 1 // ax = 1

add eax, ecx // ax = 1y

xchg eax, ebx // ax = 11y 7x - 2z, bx = 1y

idiv ebx // ax = (11y + 7x - 2z) / (1y)

mov f, eax // f = ax

}

std::cout << "f = " << f << endl;

}

1. **Результати роботи програми**

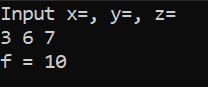


Рисунок 1. Результат роботи при значеннях x = 3, y = 6, z = 7

Перевіримо значення:

f = (11y + 7x – 2z) / (y + 1);

f = (11 \* 6 + 7 \* 3 – 2 \* 7) / (6 + 1);

f = (66 + 21 – 14) / 7;

f = 73 / 7 = 10.42;

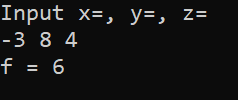


Рисунок 2. Результат роботи при значеннях x = -3, y = 8, z = 4

Перевіримо значення:

f = (11y + 7x – 2z) / (y + 1);

f = (88 – 21 – 8) / 9;

f = 59 / 9 = 6.55;

1. **Висновки**

На цій лабораторній роботі було ознайомлено з регістрами на мові Асемблер та з основними командами мови програмування, а саме: mov, xchg, add, sum, imul, idiv. Також було ознайомлено з програмуванням лінійних процесів мовою Асемблер та використання асемблерної вставки в мові C++.

1. **Відповіді на контрольні питання**
2. Вкажіть з якими числами працюють команди ***add*** та ***sub***?

Команди ***add*** та ***sub*** працюють як з числами розміром у байт, так і з числами розміром у слово. Неможна, щоб один з операндів був байтом, а інший - словом.

1. Вкажіть для чого потрібна команда ***xchg***?

Команда ***xchg*** використовується для обміну значень між регістрами.

Наприклад регістр ax = 7 та регістр bx = 2.

Після виконання команди xchg ax, bx значень регістрів будуть наступні: ax = 2, bx = 7.

1. Вкажіть у чому відмінність команд ***mov*** та ***xchg***?

***mov*** – це команда передачі значень, тобто mov ax, 2 буде означати, що в регістр ax буде записано значення 2.

***xchg*** – це команда для обміну значень між регістрами.