Лабораторна робота 1
Програмування операцій з елементами рядків цілих чисел

Архітектура IA32
Інструментальні засоби - MASM, x32Dbg, OllyDbg
Час виконання – 4 академічних години
Формування звіту
Захист із демонстрацією результатів

Мета роботи і постановка задачі

Мета роботи:
набуття практичних навичок розробки і налагодження програм на мові асемблера для програмування комп'ютерних операцій із цілими числами різної розрядності.

Початкові дані:
- рядки даних (одномірні масиви, вектор);
- кількість елементів у рядках - однакова;
- тип елементів даних – цілі.

Необхідно: З урахуванням вимог до алгоритму розробити на асемблері програму пересилання і модифікації елементів із початкових рядків та збереження результатів перетворення або аналізу в інших рядках.

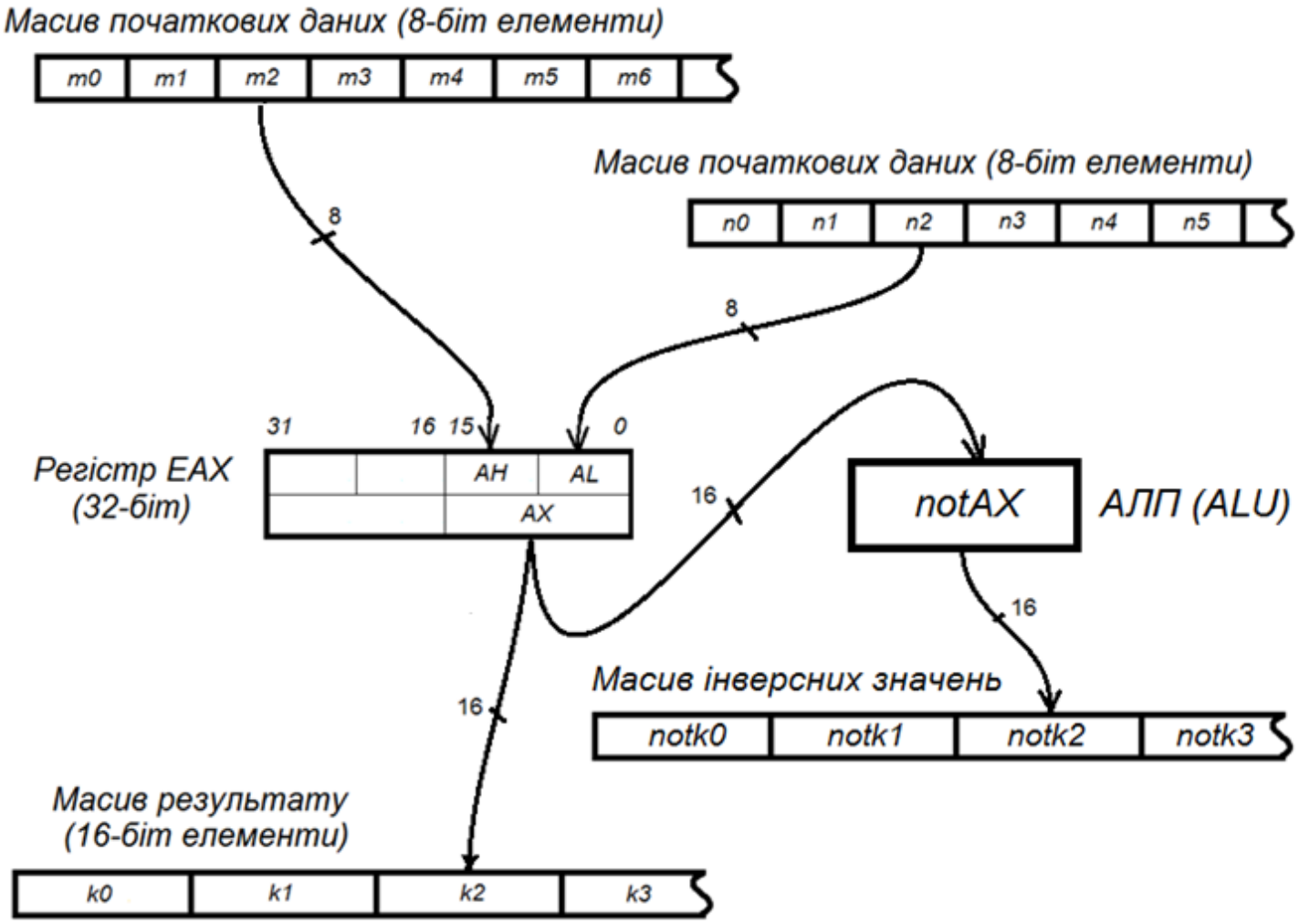
Приклад вимог до алгоритму за варіантом

- Кількість початкових рядків – 2;
- Кількість рядків результату – 2;
- Розрядність елементів (чисел) в початкових рядках - 1 байт (8 біт);
- Розрядність елементів (чисел) в рядках результату - 2 байти (16 біт);
- Дані для формування - елементи початкових рядків;
- Правило формування - конкатенація (попарне зчеплення) елементів з однаковими індексами;
- Модифікація коду - інвертування.

Зміст звіту

1. Постановка задачі.
2. Вимоги до алгоритму.
3. Графічне пояснення послідовності дій.
4. Лістинг програми з коментарем та описом роботи.
5. Print screen екрана налагоджувача з програмою.
6. Графічне і текстове пояснення результатів роботи програми.
7. Висновки за результатами виконання лабораторної роботи.

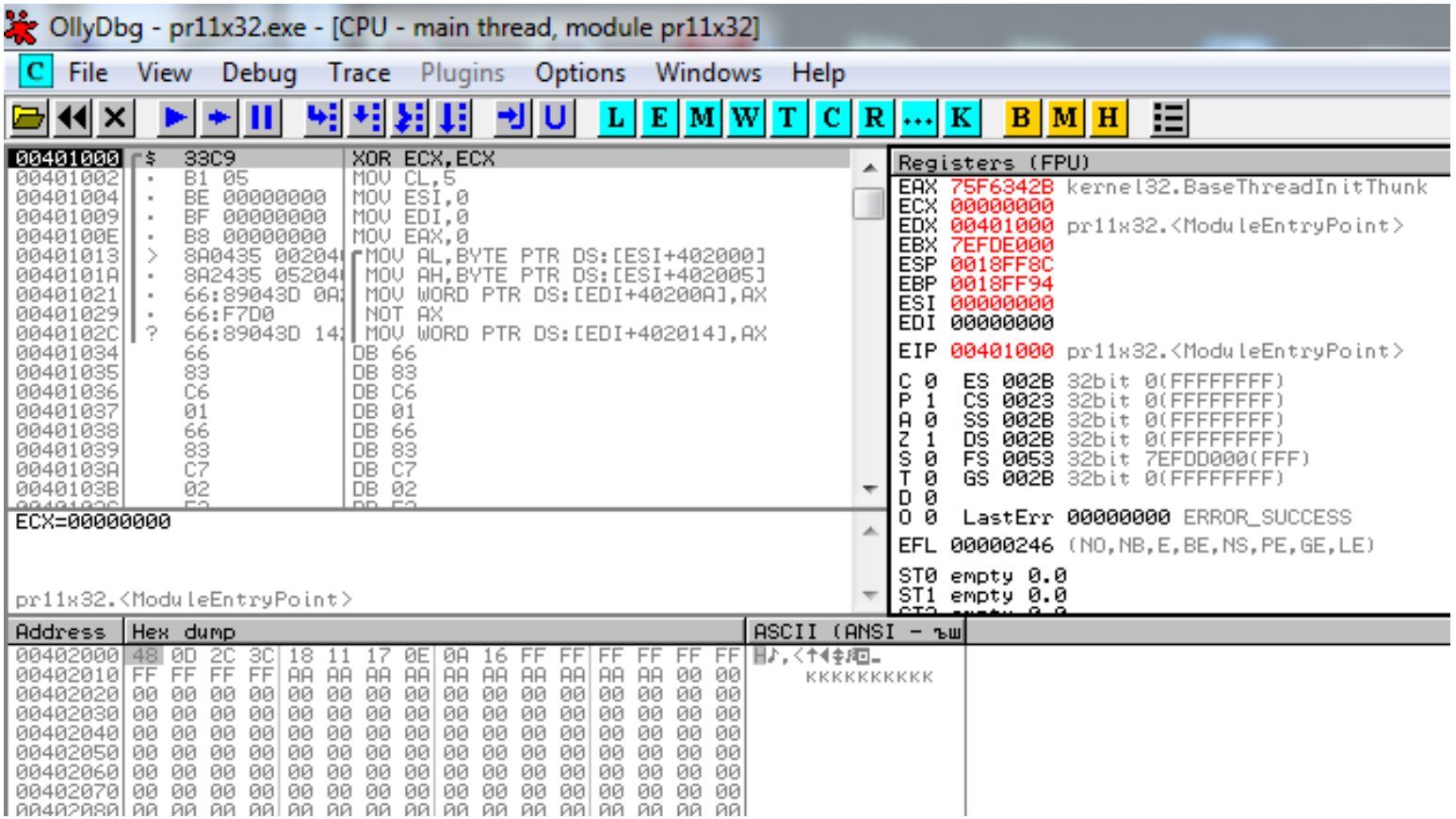
Приклад графічного пояснення дій



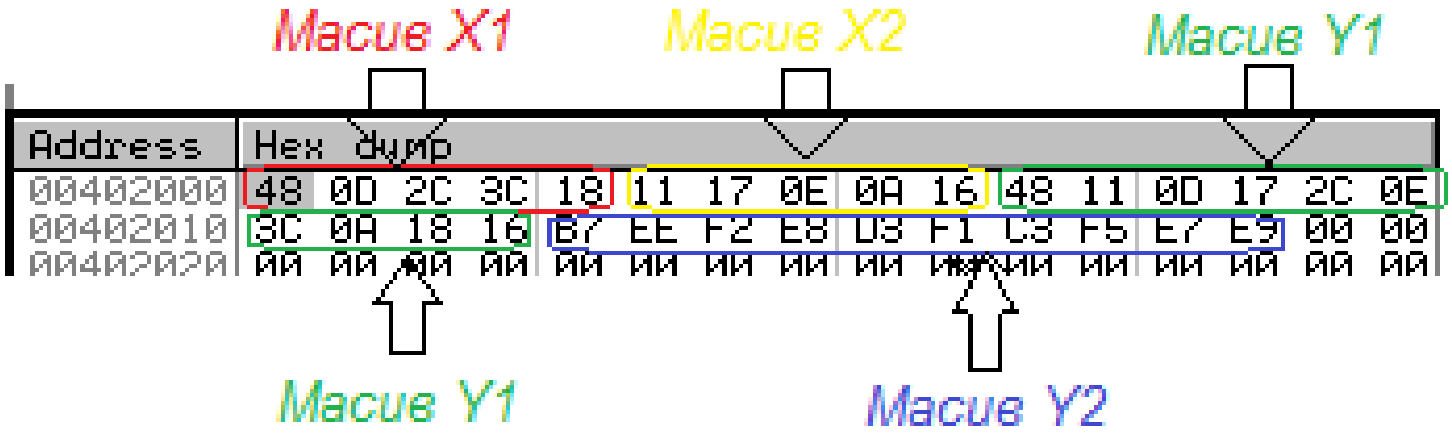
Приклад програми

```
TITLE  <Перенесення і модифікація даних>
.686
.model flat, stdcall
option casemap: none
.data
   X1 db 72, 13, 44, 60, 24
   LENX1 EQU $ -X1 ; визначення довжини масиву X1 (в байтах !!!)
   X2  db   17,  23, 14, 10, 22
   LENX2 EQU $ -X2 ; визначення довжини масиву X2 (в байтах !!!)
   Y1 dw LENX2 DUP(0FFFFh)   ; занесення одиниць до пам'ятті-приймача
   Y2 dw LENX2 DUP(0AAAAh)   ; занесення контрольного коду до пам'ятті-приймача
.code
start:
    xor ecx, ecx
    mov cl, LENX2  ; завантаження кількості чисел в регистр-лічильник
    mov esi,0
    mov edi,0
    mov eax,0
mt1: mov al, [X1+si]  ; завантаження в регістр числа (8 біт) із масиву X1
    mov ah, [X2+si]  ; завантаження в регістр числа (8 біт) із масиву X2
    mov [Y1+di], ax  ; занесення в масив Y1 числа (16 біт) із регістру
    not ax          ; інвертування коду в регістрі (16 біт)
    mov [Y2+di], ax  ; занесення в масив Y2 числа (16 біт) із регістру
    add si,1      ; зміна індексу для початкових масивів
    add di,2      ; зміна індексу для масиву результату
    loop mt1  ; зменшення (-1) лічильника і повторення при ECX/=0
 ret
end start
```

Стан перед виконанням програми

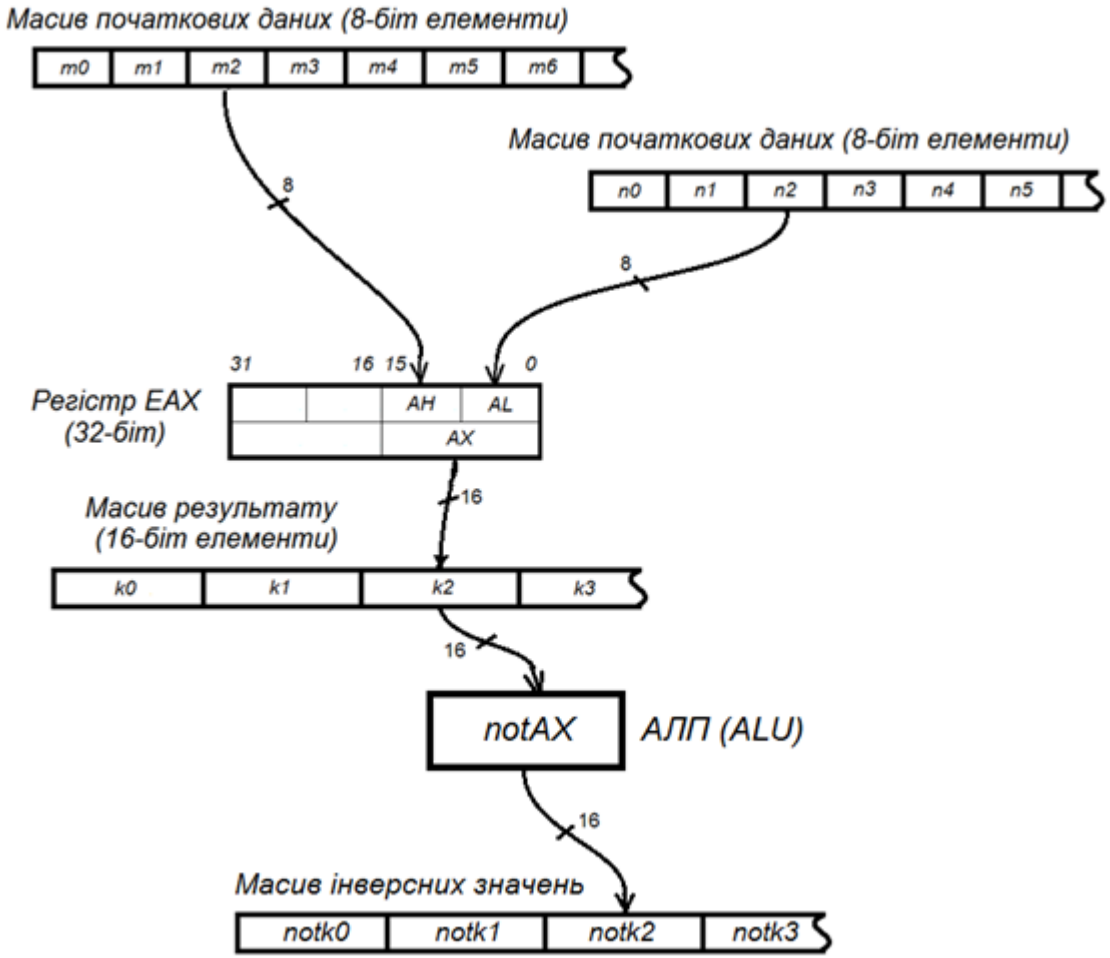


Вміст пам’яті за результатом виконання програми і контрольні показники



Прямий код = 1148, 170D, 0E2C, 0A3C, 1618
Інверсний код = EEB7, E8F2, F1D3, F5C3, E9E7

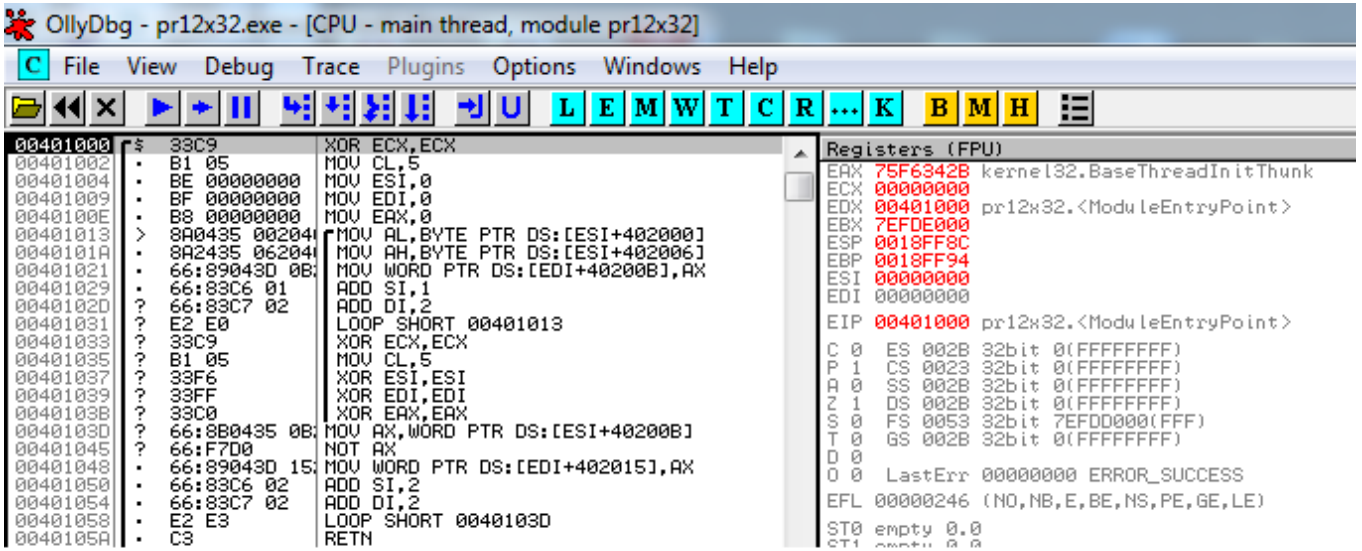
Приклад 2 графічного пояснення дій



Приклад програми 2

```
TITLE  <Перенесення і модифікація даних>
.686
.model flat, stdcall
option casemap: none
.data
   X1 db 72, 13, 44, 60, 24, 60
   LENX1 EQU $ -X1 ; визначення довжини масиву X1 (в байтах !!!)
   X2  db   17,  23, 14, 10, 22
   LENX2 EQU $ -X2 ; визначення довжини масиву X2 (в байтах !!!)
   Y1 dw LENX2 DUP(0FFFFh)   ; занесення одиниць до пам'ятті-приймача
   Y2 dw LENX2 DUP(0AAAAh)   ; занесення контрольного коду до пам'ятті-приймача
.code
start:
    xor ecx, ecx
    mov cl, LENX2  ; завантаження кількості чисел в регистр-лічильник
    mov esi,0
    mov edi,0
    mov eax,0
mt1: mov al, [X1+si]  ; завантаження в регістр числа (8 біт) із масиву X1
    mov ah, [X2+si]  ; завантаження в регістр числа (8 біт) із масиву X2
    mov [Y1+di], ax  ; занесення в масив Y1 числа (16 біт) із регістру
    add si,1      ; зміна індексу для початкових масивів
    add di,2      ; зміна індексу для масиву результату
    loop mt1  ; зменшення (-1) лічильника і повторення при ECX/=0
    xor ecx, ecx
    mov cl, LENX2  ; завантаження кількості чисел в регистр-лічильник
    xor esi, esi
    xor edi, edi
    xor eax, eax
mt2: mov ax, [Y1+si]  ; завантаження в регістр числа (16 біт) із масиву Y1
    not ax              ; інвертування коду в регістрі (16 біт)
    mov [Y2+di], ax  ; занесення в масив Y2 числа (16 біт) із регістру
    add si,2      ; зміна індексу
    add di,2      ; зміна індексу
    loop mt2  ; зменшення (-1) лічильника і повторення при ECX/=0
   ret
end start
```

Вміст пам’яті (програма 2)



До виконання програми:

Address	Hex dump															
00402000	48	0D	2C	3C	18	3C	11	17	0E	0A	16	FF	FF	FF	FF	FF
00402010	FF	FF	FF	FF	FF	AA	AA	AA	AA	AA	AA	AA	AA	AA	AA	00
00402020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00402030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00402040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Після виконанння програми:

Address	Hex dump															
00402000	48	0D	2C	3C	18	3C	11	17	0E	0A	16	48	11	0D	17	2C
00402010	0E	3C	0A	18	16	B7	EE	F2	E8	D3	F1	C3	F5	E7	E9	00
00402020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00402030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00402040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Рекомендації студентам

1. Після отримання завдання необхідно, в першу чергу, розробити та узгодити з викладачем «графічне пояснення дій».
2. Первинне налагоджування програми виконувати з мінімально необхідною кількістю даних.
3. Після отримання первинного робочого варіанта програми (або її частини) бажано остаточно узгодити з викладачем вимоги до завдання.

Література

Навчально-методичні матеріали попередніх лекцій.