МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут комп'ютерних наук та інформаційних технологій
Кафедра програмного забезпечення

ЗВІТ

до лабораторної роботи №8
на тему: «Використання функцій BIOS для роботи з відео в текстовому та графічному режимах»
з дисципліни: «Архітектура комп'ютера»

Лектор:
доц. кафедри ПЗ
Крук О. Г.

Виконав:
ст. гр. ПЗ-22
Чаус О. М.

Прийняв:
доц. кафедри ПЗ
Крук О. Г.

« ____ » _____ 2022 р.

∑= _____ .....:

Львів-2022

Тема роботи: Використання функцій BIOS для роботи з відео в текстовому та графічному режимах.

Мета роботи: опанувати функції BIOS для роботи з відео в текстовому та графічному режимах; розвинути навики складання програм для виведення різнокольорових рядків символів та графічних зображень; відтранслювати і виконати в режимі відлагодження програми, складені відповідно до свого варіанту.

Варіант 27

Теоретичні відомості

Функції BIOS переривання INT 10h для роботи з відео в текстовому режимі

У текстовому режимі прикладна програма може вивести інформацію на екран одним з таких способів.

• За допомогою функцій MS DOS. Якщо на комп'ютері встановлена система MS DOS або її емулятор, для виведення текстових даних на екран можна скористатися функціями переривання INT 21h. Дані функції дозволяють перенаправити потоки введення-виведення на будь-який інший пристрій, такий як принтер або диск. Виведення на екран за допомогою функцій переривання INT 21h виконується досить повільно, і колір символів змінити не можна.

• За допомогою функцій BIOS. Вивести символи на екран можна також за допомогою функцій переривання INT 10h, оброблення якого виконується системою BIOS, а не DOS. Вони виконуються набагато швидше, ніж функції переривання INT 21h, і дозволяють змінити колір тексту на екрані. При заповненні символами великих областей на екрані за допомогою функцій переривання INT 10h, можна помітити невелику затримку виведення. Крім того, дані, що виводяться на екран, не можна перенаправити на інший пристрій.

• Прямий доступ в відеопам'ять. Вивести символи на екран можна також шляхом переміщення їх безпосередньо в область пам'яті відеоадаптера. При цьому досягається максимальна швидкість виведення, проте дані також можна перенаправити на інший пристрій. На зорі розвитку ПК, коли основною операційною системою була MS DOS, в прикладних програмах, таких як текстові процесори і електронні таблиці, використовувався саме цей метод виведення даних на екран. Слід зазначити, що даний метод також можна використовувати при роботі програми в повноекранному режимі під керуванням операційних систем Windows NT, 2000 і XP.

Залежно від поставлених завдань, в застосунку може використовуватися один з трьох запропонованих вище способів виведення даних на екран. Якщо на перше місце ставиться швидкість виведення на екран, то потрібно скористатися прямим виведенням у відеопам'ять. В інших випадках слід віддати перевагу функціям BIOS. Функціями DOS варто користуватися тільки тоді, коли вихідний потік даних може бути перенаправлений на інший пристрій або коли екран спільно використовується кількома програмами. Слід зазначити, що для виведення даних на екран у функціях MS DOS використовуються функції BIOS, а у функціях BIOS - прямий доступ до відеопам'яті.

Відеоадаптер комп'ютера може працювати в двох режимах: текстовому і графічному. Під час початкового завантаження MS DOS відеоадаптер переводиться в текстовий режим роботи і встановлюється відеорежим номер 3 (кольоровий текстовий режим, 25 рядків і 80 стовпців). Однак крім текстового, існує і ряд графічних відеорежимів, частину з яких перераховано в табл. 3.

У текстовому режимі рядка нумеруються з нуля зверху вниз екрану. Висота кожного рядка визначає розмір відображуваних на екрані символів і залежить від поточного використовуваного шрифту. Стовпці екрану нумеруються з нуля зліва направо. Ширина кожного стовпця, як і висота рядка, також визначає розмір відображуваних на екрані символів і також залежить від поточного використовуваного шрифту.
Шрифти. Зовнішній вигляд символів, що відображаються на екрані, визначається спеціальною таблицею, яка перебуває в пам'яті і містить образи символів шрифту. У перших версіях BIOS ця таблиця розташовувалася в ПЗУ, однак з часом в BIOS з'явилися функції, завдяки яким прикладна програма під час виконання може завантажити з пам'яті власний шрифт. Це стимулювало користувачів до розробки оригінальних шрифтів для текстового режиму роботи відеоадаптера.
Текстові відеосторінки.
Текстова пам'ять містить 8 відеосторінок і займає в адресному просторі комп'ютера (за межами звичайної пам'яті) 32 Кбайти від сегментної адреси B800h. Починається вона з відео 0, адреса якої збігається з адресою всієї відеопам'яті. Кожна сторінка займає 4 Кбайт; таким чином, сторінка 1 починається з сегментної адреси B900h, сторінка 2 - з адреси BAOOh тощо.
При включенні комп'ютера активною (видимою) стає відеосторінка 0. Зміна відеосторінок здійснюється викликом функції 05h переривання 10h BIOS.
Таким чином, у програми з'являється можливість під час відображення однієї сторінки виводити дані в іншу приховану відеосторінку, а потім швидко перемикнути вміст екрану. Раніше при створенні високопродуктивних застосунків для MS DOS часто доводилося зберігати в пам'яті відразу кілька копій текстових екранів. В даний час особливою популярністю користуються програми з графічним інтерфейсом, тому текстові відеосторінки втратили свою значимість. За замовчуванням використовується нульова відеосторінка.
Атрибути. Кожному символу, що відображається на екрані, призначається спеціальний байт атрибутів, який визначає його колір, а також колір розташованого за ним фону.
Кожній позиції екрану відповідає один символ і один атрибут кольору. Значення атрибута зберігається в окремому байті, який у відеопам'яті розташований відразу ж за символом. На рис. 1 показано розташування у відеопам'яті трьох символів "ABC", а також їх атрибутів.

Зміна кольорів

Змішування основних кольорів
Колір кожного пікселя зображення визначається значенням струму трьох незалежних променів електронно-променевої трубки монітора: червоного, зеленого і синього. Існує ще один, четвертий канал керування кольором, який змінює загальну інтенсивність (тобто яскравість) всіх пікселів. Таким чином, значення всіх доступних кольорів в текстовому режимі можна визначити у вигляді одного 4-бітового двійкового числа, заданого в наступному форматі: I = інтенсивність, R = червоний, G = зелений, B = синій.
Функція 13h переривання INT 10h дозволяє вивести текстовий рядок на екран з вказаної у вигляді номера рядка і стовпця позиції. У рядку, крім символів, можуть бути вказані і байти атрибутів.

INT 10h, функція 13h
Опис - виводить рядок на екран в режимі емуляції телетайпа.
Параметри:
AH = 13h
AL = режим записування
BH = номер відеосторінки
BL = байт атрибутів (якщо AL = 00h або 01h)
CX = довжина рядка (лічильник символів)
DH, DL = номер рядка і стовпця
ES:BP = адреса рядка в формі "сегмент-зміщення"
В регістрі AL можна задати наступні режими записування:
• 00h - в заданому рядку вказані тільки ASCII-коди символів; після виведення на екран поточне положення курсора не змінюється; в регістрі BL вказаний байт атрибутів;
• 01h - в заданому рядку вказані тільки ASCII-коди символів; після виведення на екран поточне положення курсора коригується з урахуванням довжини рядка; в регістрі BL вказано байт атрибутів;
• 02h - в заданому рядку вказані ASCII-коди символів, слідом за якими упереміш розташовані байти атрибутів; після виведення на екран поточне положення курсора не змінюється;
• 03h - в заданому рядку вказані ASCII-коди символів, слідом за якими упереміш розташовані байти атрибутів; після виведення на екран поточне положення курсора коригується з урахуванням довжини рядка.
Відображення графіки за допомогою безпосереднього записування у відеопам'ять

Основний недолік описаного способу виведення пікселів на екран за допомогою однієї з функцій переривання INT 10h полягає у дуже повільній швидкості виведення. Справа в тому, що для виведення одного пікселя щоразу доводиться викликати переривання INT 10h, на оброблення якого операційна система витрачає багато часу. Набагато ефективніший метод відображення пікселів на екрані полягає у безпосередньому їх записуванні у відеопам'ять. Ця методика зазвичай називається прямим доступом до відеопам'яті.

Відеорежим 13h: 320x200, 256 кольорів

Найзручнішим відеорежимом під час використання прямого доступу до відеопам'яті є режим 13h. При його встановленні кожен піксель екрану займає один байт у відеопам'яті, яка подається у вигляді двовимірного масиву байтів, що відповідають рядкам та стовпцям зображення. Піксель, розташований у лівому верхньому кутку екрана, відповідає першому байту масиву, що має нульове зміщення. У першому рядку екрану міститься 320 пікселів, які відповідають першим 320 байтам масиву відеопам'яті. Наступні 320 байтів масиву відповідають пікселям другого рядка екрану тощо. Останньому байту масиву відповідає піксель, розташований у правому нижньому куті екрану. Чому кожному пікселю відведено цілий байт? Вся справа в тому, що для подання 256 різних кольорів потрібно 256 значень цілих чисел, що відповідає 8 бітам або одному байту.
Команда OUT. Керування роботою відеоадаптера здійснюється програмно за допомогою команд OUT (Output to port або Виведення в порт). З їх допомогою

встановлюється колірний режим, роздільна здатність екрану та палітра кольорів. Перед виконанням команди OUT в регістр DX завантажується 16-розрядна адреса порту, а в регістр AL - значення, що виводиться в порт. До прикладу, регістр, який керує палітрою кольорів, має адресу порту 3C8h. У наведеному нижче фрагменті програми в цей порт виводиться значение 20h.

```
mov dx, 3C8h        ; Адреса порту
mov al, 20h         ; Значення, що виводиться
out dx, al          ; Команда записування в порт
```

Індекси кольору. Одна з особливостей використання відеорежиму номер 13h полягає в тому, що цілі числа, що визначають один із 256 кольорів пікселів, не прямо, а опосередковано впливають на їх колір. Насправді ці числа є індексами, за значенням яких вибирається реальний колір зі спеціальної таблиці кольорів, яка називається палітрою (palette). Кожен елемент палітри кольорів складається із трьох цілих чисел, значення яких міститься в діапазоні 0-63. Ці числа визначають інтенсивність кожного із трьох променів: червоного, синього та зеленого (RGB). Нульовий елемент кольорової палітри визначає колір фону екрана.

Таким чином, використовуючи палітру кольорів, можна створити 262144, або 643 різних кольорів. Однак тільки 256 з них можуть одночасно відображатися на екрані. Тим не менш, у програмі можна швидко перемикнути палітру кольорів і таким чином впливати на кольори, що відображаються на екрані.

Кольори RGB. При формуванні RGB-кольорів використовується адитивна колірна модель, при якій яскраво-білий колір отримується шляхом змішування всіх трьох кольорів в рівних пропорціях. Крім адитивної, використовується також субтрактивна модель кольору, яка використовується при кольоровому друкуванні на папері. У ній кольори утворюються шляхом віднімання з яскраво-білого кольору одного з основних кольорів.

При використанні адитивної колірної моделі чорний колір виходить за повної відсутності колірних складових. Для отримання білого кольору необхідно встановити максимальну інтенсивність основних колірних складових, тобто. присвоїти їм у палітрі кольорів значення 63. Якщо одночасно змінювати значення складових всіх трьох кольорів, то отримаються різні відтінки сірого кольору.

Для отримання чистих кольорів необхідно значення всіх складових кольорів, крім однієї, встановити в нуль. Щоб отримати світлі відтінки будь-якого кольору, необхідно збільшити в рівних пропорціях складові двох інших кольорів.

Відтінки двох інших кольорів (синього та зеленого) отримуються за аналогією з червоним. Для отримання інших кольорів, таких як фіолетовий або жовтий, доведеться змішати в різних пропорціях основні кольори.

Хід роботи

1. Склав програму для виведення тексту відповідно до завдання.
   Текст програми:

```
.model small
.386
.stack
.data
surname BYTE 'C',63h,'h',70h,'a',20h,'u',1Fh,'s',17h
s_name BYTE 'O', 3Ch, 'l', 5Eh, 'e', 2Bh, 'h', 112
patronymic BYTE 'M', 64h, 'y', 1Bh, 'k', 62h, 'o', 4h, 'l', 40h, 'a', 2h,
'i', 43h, 'o', 2Bh, 'v', 24h, 'y', 65h, 'c', 2Eh, 'h', 7h
row DWORD 2
column DWORD 49
.code
main PROC
    call ClrScr
    mov AX, seg surname
    mov ES, AX
    mov AH, 13h ;функція виводу рядка
    mov AL, 2 ;режим запису з кольорами, курсор не змінюється
    mov BH, 0 ;сторінка
    mov CX, (SIZEOF surname) / 2
    mov DH, 7
    mov DL, 49 ;рядок та стовбець
    mov BP, offset surname
    int 10h
    mov AX, seg s_name
    mov ES, AX
    mov AH, 13h ;функція виводу рядка
    mov AL, 2 ;режим запису з кольорами, курсор не змінюється
    mov BH, 0 ;сторінка
    mov CX, (SIZEOF s_name) / 2
    mov DH, 9
    mov DL, 52 ;рядок та стовбець
    mov BP, offset s_name
    int 10h
    mov AX, seg patronymic
    mov ES, AX
    mov AH, 13h ;функція виводу рядка
    mov AL, 2 ;режим запису з кольорами, курсор не змінюється
    mov BH, 0 ;сторінка
    mov CX, (SIZEOF patronymic) / 2
    mov DH, 13
    mov DL, 57 ;рядок та стовбець
    mov BP, offset patronymic
    int 10h
    mov ah,10h
    int 16h
    .exit
main ENDP

ClrScr:
mov al, 3
int 10h
ret
END main
```

Результат виконання програми:

2. Склав програму для виведення прямокутника відповідно до індивідуального завдання.

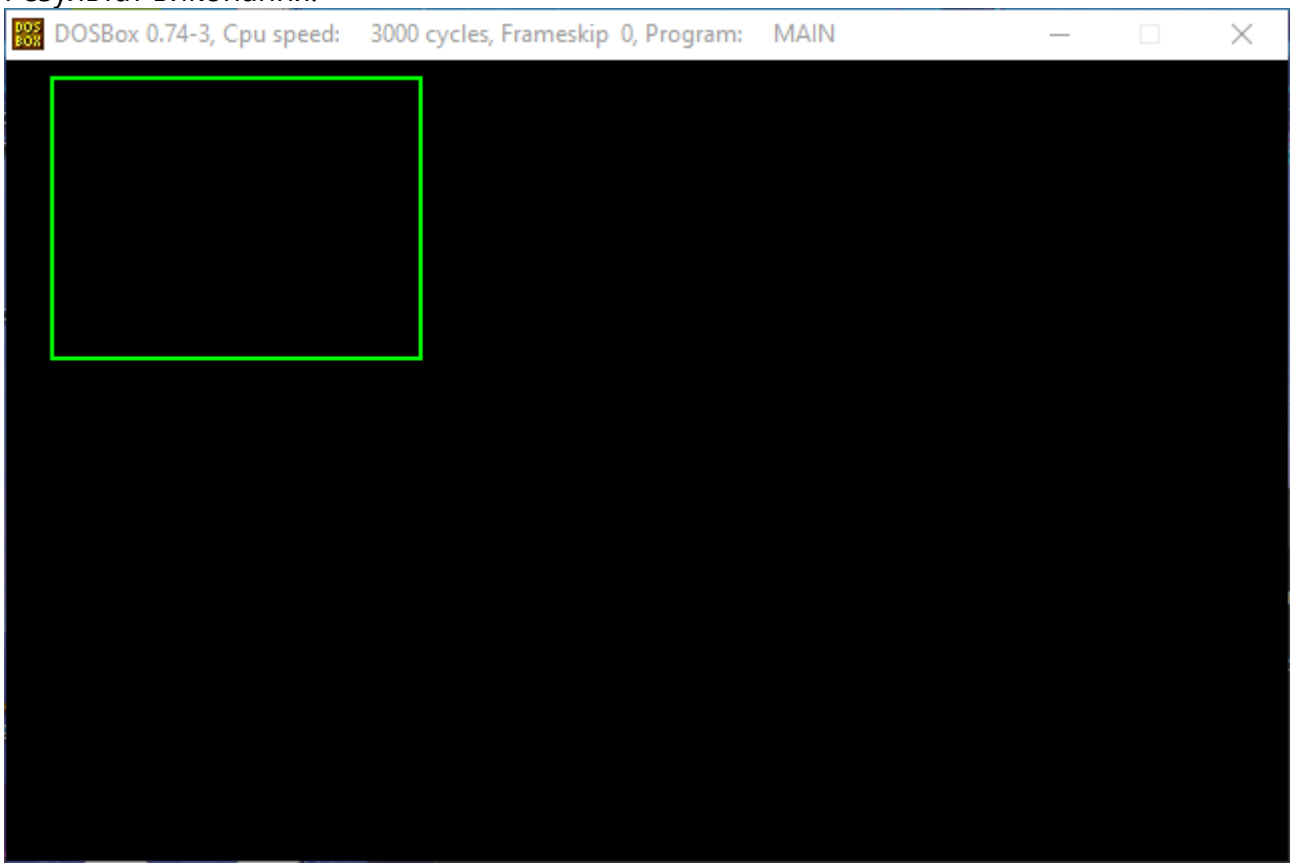
Текст програми:

```
INCLUDE Irvine16.inc
.data
horizontal_length WORD 93
vertical_length WORD 71
color WORD 2
saveMode BYTE ?
xVal        WORD    11
yVal        WORD    4
.code
main PROC
    mov ax, @data
    mov ds, ax
    ;----------
    ;збереження попереднього відеорежиму
    mov ah, 0Fh
    int 10h
    mov saveMode, al
    ;----------
    ;встановлення графічного режиму
    mov ah, 0
    mov al, 13h
    int 10h
    ;----------
    ;завантаження адреси відеобуфера
    push 0A000h
    pop es
```

```
;----------
;встановлення палітри
mov dx, 3c8h
mov al, 1
out dx, al
mov dx, 3c9h
mov al, 0
out dx, al
mov al, 63
out dx, al
mov al, 0
out dx, al

;будування лівої сторони
mov ax, 320
mul yVal
add ax, xVal
mov di, ax
mov cx, vertical_length
l1:
    mov es:[di], 1
    add di, 320
    loop l1

;будування верхньої сторони
mov ax, 320
mul yVal
add ax, xVal

mov cx, horizontal_length
mov di, ax

l2:
    mov es:[di], 1
    add di, 1
    loop l2

;будування нижньої сторони
mov bx, xVal
add bx, horizontal_length
dec bx
mov ax, 320
mul yVal
add ax, bx
mov di, ax
mov cx, vertical_length
l3:
    mov es:[di], 1
    add di, 320
    loop l3

;будування правої сторони
mov bx, yVal
```

```
    add bx, vertical_length
    dec bx
    mov ax, 320
    mul bx
    add ax, xVal
    mov di, ax
    mov cx, horizontal_length
    l4:
        mov es:[di], 1
        add di, 1
        loop l4

    ;очікування на ввід користувача
    mov ax, 0003h
    int 16h
    ;встановлення попереднього відеорежиму
    mov ah, 0
    mov al, saveMode
    int 10h

 main ENDP
END main
```

Результат виконання:



Висновки: на цій лабораторній роботі я опанував функції BIOS для роботи з відео в текстовому та графічному режимах; розвинув навики складання програм для виведення різнокольорових рядків символів та графічних зображень; відтранслював виконати в режимі відлагодження програми, складені відповідно до свого варіанту.