Лабораторна робота №5

ВИВЕДЕННЯ ДАТИ ТА ЧАСУ НА ЕКРАН

Мета: Ми продовжуємо знайомитись з функціями виводу на екран та з функціями роботи з системним таймером. В цій лабораторній роботі ми вчимося звертатися до функцій DOS та BIOS, що повертають дату та час. Ми також вчимося виводити числові данні на екран, користуючись знайомими вже функціями виводу.

Обладнання: будь-який текстовий редактор, компілятор асемблера (TASM або MASM), дебагер (рекомендується Turbo Debuger).

Хід роботи

1. Постановка задачі та розробка алгоритму.

За умовою нашої лабораторної роботи ми повинні зчитати за допомогою функцій DOS та BIOS данні про поточний час та дату і вивести ці данні у вигляді рядка на екран.

Рядок, що ми отримаємо на екрані повинен мати наступну структуру:

1. Поточний день (число).
2. Місяць, записаний словом.
3. Рік (число).
4. День тижня (слово).
5. Поточна година (число).
6. Хвилини (число).
7. Секунди (число).

Слід пам’ятати, що від функцій DOS та BIOS данні про поточний час та дату ми отримаємо у вигляді чисел, а от вивести на екран ми можемо лише рядок, що складається з ASCII символів. Тому перед нами стоїть дві задачі: для місяця і дня тижня замінити число на рядок з назвою цього місяця або дня тижня, а для всіх інших даних – перевести їх з числової форми у вигляд символьного рядку, що буде містити те саме число у вигляді ASCII символів.

Місяць та день тижня нам найкраще отримати за допомогою функцій DOS. По-перше, тільки функції DOS повертають нам номер дня тижня, а по-друге, функції DOS повертають значення поточного місяця і дня тижня у вигляді звичайних двоїчних чисел, з якими легко організовувати перевірки та умовні переходи і підставляти замість них символьні рядки з назвами місяця та дня тижня.

Всі інші дані нам найкраще отримати за допомогою функцій BIOS, тому що результати цих функцій повертаються у двійково-десятичній формі, яку легко обробити і перетворити в символьний рядок. Для цього досить розділити між собою 4 молодших та 4 старших біти кожного байту, збільшити кожне отримане значення на 48 (код символу 0) і ми отримаємо ASCII код того числа, яке ми шукали.

Стає ясним, що в нашій програмі основні складності будуть з організацією умовних переходів та з перетворенням чисел з двійково- десятичної форми в ASCII-формат.

В загальному вигляді алгоритм цієї задачі буде такий:

1. Зчитуємо дату за допомогою функції DOS.
2. Запам'ятовуємо місяць та день тижня.
3. Зчитуємо дату за допомогою функції BIOS.
4. Перетворюємо номер дня в рядок.
5. Виводимо день на екран.
6. Замість номеру місяця виводимо його назву.
7. Перетворюємо номер року в рядок.
8. Виводимо рік на екран.
9. Замість номеру дня тижня, виводимо назву дня тижня.
10. Перетворюємо номер години в рядок.
11. Виводимо години на екран.
12. Перетворюємо номер хвилини в рядок.
13. Виводимо хвилини на екран.
14. Перетворюємо секунди в рядок.
15. Виводимо секунди на екран.

Цей алгоритм можна зробити набагато компактнішим, якщо замість виводу на екран кожного елементу окремо, сформувати спочатку повний рядок з усіх потрібних елементів, і вже наприкінці програми вивести його на екран.

За отриманим алгоритмом студенти повинні побудувати блок-схему та скласти програму.

2. Написання програми.

Скориставшись будь-яким текстовим редактором, що не використовує службових символів, необхідно за складеним алгоритмом написати програму на мові асемблера.

Для зчитування номеру місяця та дня тижня нам знадобиться функція DOS 2Ah (INT 21h). Отримавши у вигляді числа ці значення ми можемо легко підставити замість них заздалегідь заготовані рядки з назвами місяців та днів тижнів.

Для зчитування всіх інших даних краще користуватися функціями BIOS. Так для зчитування дати нам знадобиться функція 04h (INT 1Ah), а дня зчитування поточного часу знадобиться функція 02h (INT 1Ah).

№ в AH	Описание	Вход	Выход
2Ah	Отримати системну дату		AL = день тижня (0-неділя...) DH = місяць (число починаючи з 1)
04h	Читати дату		DL = число місяця вкоді BCD (приклад dx: 0312h = 12 березня)
02h	Читати час		AX = година, хвилина в коді BCD (пд. CX=1243h = 12:43) DH= секунди в коді BCD

Функція 2Ah повертає дані в шістнадцятковій формі. Наприклад, для того щоб вивести день тижня необхідно наперед підготувати строку в пам'яті із переліченими днями, вони мають мати однакову довжину, і після кожного дня потрібно записати в пам'ять знак кінця строки.

Отримавши у відповідному регістрі число дня ми множимо на довжину імені дня, таким способом ми отримаємо адрес зміщеня відповідного дня в цій строці, додавши його до адреса строки, отримаємо адрес зміщення в пам’яті, після чого можемо вивести його на екран (виведення буде до першого символа кінця строки, який ми мали заздалегідь записали в пам’ять)

Функція 04h і 02h повертає дані в двійково-десятковій формі.

Переклад чисел у символьну форму 16-річной системи. При створенні найрізноманітніших програмних продуктів часто потрібно виводити на екран які-небудь числа. При цьому виникає необхідність перетворення цих чисел із внутрішнього двійкового формату, у якому вони обробляються ЕОМ, у символьний, у якому вони можуть бути відображені на екрані. Для представлення чисел використовуються різні системи числення. Для людини звична десяткова система. Однак при висновку вмісту осередків оперативної пам'яті, адрес і інших системних значень частіше використовується 16-ричная система числення, тому що її використання в цьому випадку більш природно. Крім того, перетворення в 16-ричную систему здійснюється простіше, ніж у десяткову.

Переклад чисел із двійкової системи числення в 16-річну здійснюється в такий спосіб: розряди двійкового числа групуються по 4, починаючи з молодшого, після чого кожна четвірка розрядів (тетрада) перетвориться у відповідну 16-річну цифру (у нашому випадку — у ASCII-код цієї цифри).

У процесорі 8086 немає засобів звертання до четвірок бітів, мінімальним осередком оперативної пам'яті, до якого він має доступ, є байт. Тому необхідно кожну тетраду розширити до байта і лише потім перетворювати результат у ASCII-код.

Для виділення значень окремих бітів двійкового числа застосовується операція логічного множення (AND) по масці. Наприклад, для видділенні другої по старшинству тетради двійкового слова (припустимо, регістра AX) використовується маска 0F0h, наприклад:

```
AN      1010 1011 1100 1101b    =       ABCDh (вихідне число в AX)
AN      0000 0000 1111 0000b    =       00F0h (маска)
        0000 0000 1100 0000b    =       00C0h (результат);
```

відповідна асемблерна команда:

```
and    ax, 0F0h
```

Для зрушення сукупності бітів щодо розрядної сітки використовуються команди логічного зрушення (зокрема, SHR — зрушення вправо). Щоб зрушити цифру 3, отриману вище, можна використовувати команду

```
shr    ax, 4        ; зрушення вправо на 4 розряди
```

Після того, як тетрада підготовлена, необхідно перетворити її в ASCII- код. Одним з методів є використання таблиці перетворення, що містить ASCII-коди преутворених цифр. При цьому номер осередку таблиці, що відповідає одній цифрі, заміняється значенням даного осередку —

кодом цифри. Таблиця перетворення (трансляції) оформляється у виді набору символів цифр, записаних у порядку зростання:

```
tabl      db     '0123456789ABCDEF'
```

Для вибірки значення з таблиці зручно використовувати команду табличної трансляції *XLAT* (без операндів). Перед виконанням XLAT необхідно занести в регістри DS:BX повну адреса таблиці, а в AL — номер (n) осередку таблиці. XLAT поміщає в той же регістр AL значення n-й осередку таблиці, наприклад:

```
lea    bx, tabl
mov    al, 9
xlat                ; AL = '9' = 39h
```

Перетворені цифри можна заносити в пам'ять або відразу виводити на екран.

Для виведення символів ми можемо користуватися будь-якою з освоєних нами функцій виводу на екран.

Після написання програми, вона компілюються за допомогою вибраного компілятора.

3. Тестування програми. Аналіз результатів.

Після написання програми необхідно перевірити її працездатність.

За допомогою дебагера потрібно уважно крок за кроком виконати всю програму, звернути увагу на правильність адресації, при замінені числа на рядок, звернути увагу на правильність перетворення двійково-десятичного числа в рядок.

Лише після ретельного тестування і переконання в тому, що програма працює абсолютно вірно, її можна запускати не в тестовому режимі.

4. Висновки.

В цій лабораторній роботі ми ставили перед собою і виконали дві основні задачі: по-перше, ми вчились зчитувати інформацію про поточний час та дату за допомогою різних функцій (як DOS так і BIOS) – ту інформацію, що нерідко буває дуже потрібною при написанні різних програм. По-друге, ми вчились перетворювати числа в ASCII- рядки, що також буває необхідним в багатьох випадках і є важливим моментом при програмуванні на низькому рівні.

5. Контрольні запитання.

1). Які функції DOS для зчитування дати та часу ви знаєте?

2). Які функції BIOS для зчитування дати та часу ви знаєте?

3). Яка різниця між функціями DOS та BIOS для зчитування часу та дати?

4). Опишіть послідовність дій при переведенні двійково-десятичного числа в ASCII-рядок?

5). Яка була б послідовність дій, якщо нам потрібно було б перетворити звичайне число в ASCII-рядок?

Код програми:

```
model tiny
.code
.startup
        mov al,2
        mov ah,00h                          ;очистка екрана
        int 10h
;-------------------------------------------------
mov ah,04h                                  ;отримання дня місяця
int 1ah

mov si,dx                                    ;запис дня в SI
mov al,dl                                   ;запис дня в  AL
        call first                          ;перекодування першої цифри
        call sumbol                         ;вивід першої цифри

mov al,dl                                   ;аналогічні операції з другою цифрою
        call second
        call sumbol
;-------------------------------------------------------
        mov ah,2ah                          ;місяць
        int 21h                             ;отримання номера місяця

        mov al,dh                           ;запис в AL номера місяця
        sub al,1                            ;AL – 1(тому що починається з 1, а не з 0)
        mov ah,10                           ;AH=10 (довжина імені місяця)
        mul ah                              ;AH=AH*AL, AL=0
        lea dx,Mis                          ;загружаємо адрес місяця вDX
        add dx,ax                           ;DX=DX+AX -  адрес потрібного місяця

        mov ah,9h                           ;виведення місяця на екран
        int 21h
;-------------------------------------------------------
mov ah,04h              ;RIK
int 1ah

mov al,ch
        call first
        call sumbol

mov al,ch
        call second
        call sumbol
```

```
mov al,cl
      call first
      call sumbol

mov al,cl
      call second
      call sumbol

mov al,0                                    ; виведеня пробіла
call sumbol
;-------------------------------------------------------------
      mov ah,2ah                            ;день
      int 21h

      mov ah,11
      mul ah
      mov dx,Offset Den
      add dx,ax

      mov ah,9h
      int 21h
;-------------------------------------------------------------
mov ah,02h                                  ;CHAS
int 1ah
                                            ;година
mov al,ch
      call first
      call sumbol

mov al,ch
      call second
      call sumbol

mov al,58                                   ;виведення двокрапки
call sumbol


mov al,cl                                   ;хвилини
      call first
      call sumbol

mov al,cl
      call second
```

```
        call sumbol
mov al,58                                   ;виведення двокрапки
call sumbol
mov ah,02h                                  ;секунди
int 1ah
mov ch,dh
mov al,ch
        call first
        call sumbol
mov al,ch
        call second
        call sumbol
;----------------------------Procedyru---------------------
First proc                                  ;перекодування першої цифри
        and al,0f0h                         ;множимо регістр al на маску 0f0h
        shr al,4                            ;здвигаємо на 4 біти вправо
        lea bx,tabl                         ;загружаємо в BX адрес зміщення таблиці
        xlat                                ;отримуємо відповідний код символа із таблиці
                                            ;поміщений в AL
        ret                                 ;повертаємо роботу головній програмі
First endp
Second proc                                 ;процедура аналогічна попередній тільки інша
        and al,0fh                          ;маска, перекодування другої цифри  з AL
        lea bx,tabl
        xlat
        ret
Second endp
sumbol proc                                 ;вивід символа на екран, що знаходиться в AL
        lea bx,Got                          ;загрузка адреса зміщення GOT
        mov [bx],al                         ;запис по загруженому адресу зачення AL
        mov dx,bx                           ;вивід символа на екран
        mov ah,9h
        int 21h
        ret
sumbol endp
;-----------------------------------------------------------------
        RET
Got db 1 dup (' '),'$'              ;буфер для одного символа
DEN db ' Nedilya  $ Ponedilok$ Vivtorok $ Sereda   $ Chetver  $ Pyatnucya$ Subota   $'
Mis db ' Sichen  $ Lyutuy  $ Berezen $ Kviten  $ Traven  $ Cherven$ Lupen   $ Serpen  $ Veresen $
Ghovten $ Lustopad$ Gruden  $'              ; наперед підготовлені строки
tabl        db '0123456789 '                ;таблиця
end
```

Додаткові завдання за варіантами:

0. Написати на мові асемблер com-програму, яка дозволить вивести данні про перший високосний рік від поточного.
1. Написати на мові асемблер exe-програму, яка дозволить вивести данні про перший високосний рік від поточного.
2. Написати на мові асемблер com-програму, яка дозволить вивести данні про те, який місяць наступить через півроку (бажано у текстовій формі).
3. Написати на мові асемблер exe-програму, яка дозволить вивести данні про те, який місяць наступить через півроку (бажано у текстовій формі).
4. Написати на мові асемблер com-програму, яка дозволить вивести данні про те, який час буде через 1 годину і 20 хвилин (години і хвилини).
5. Написати на мові асемблер exe-програму, яка дозволить вивести данні про те, який час буде через 1 годину і 20 хвилин (години і хвилини).
6. Написати на мові асемблер com-програму, яка дозволить вивести поточну дату (число, місяць, рік) з використанням текстових форм на трьох мовах – російській, англійській і українській.
7. Написати на мові асемблер exe-програму, яка дозволить вивести поточну дату (число, місяць, рік) з використанням текстових форм на трьох мовах – російській, англійській і українській.
8. Написати на мові асемблер com-програму, яка дозволить вивести поточну дату (число, рік, день тижня) з використанням текстових форм на трьох мовах – російській, англійській і українській.
9. Написати на мові асемблер exe-програму, яка дозволить вивести поточну дату (число, рік, день тижня) з використанням текстових форм на трьох мовах – російській, англійській і українській.