UNIUNEA EUROPEANĂ
GUVERNUL ROMÂNIEI
Instrumente Structurale
2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Programare în limbaj de asamblare

44. Funcţii DOS şi BIOS.

Utilizarea funcțiilor BIOS

În timp ce BIOS-ul manipulează întreruperile INT 00H – 1FH, DOS-ul manipulează întreruperile INT 20H – 3FH. Principalele puncte de intrare ale BIOS-ului ce pot fi folosite de utilizator sunt următoarele:

- INT 00H = este de fapt o excepție de împărțire prin zero, pentru care se afişează un mesaj corespunzător;
- INT 01H = „Single Step“, este utilizată de programele de depanare (Debug, TD);
- INT 02H = întrerupere nemascabilă (NMI) care este utilizată de erori hardware severe, cum ar fi eroarea de paritate la memorie;
- INT 03H = este întreruperea de breakpoint, adică cea utilizată de programele de depanare, întrucât ocupă un singur octet şi poate substitui uşor primul octet al instrucțiunii unde se stabileşte punctul de suspendare a execuției programului (este descrisă în secțiunea dedicată întreruperilor);
- INT 04H = întreruperea de depăşire a rezultatului operațiilor aritmetice (INTO);
- INT 05H = întreruperea imprimă memoria video; aceeaşi operație poate fi activată extern (nu intern, cum o face instrucțiunea INT 05H) prin combinația de taste CTRL+PrtSc; la adresa 50:00H din zona de date BIOS se poate citi starea acestei operații;
- INT 08H = ceasul de timp real; o întrerupere hard actualizează ora şi data sistemului (un circuit de ceas programabil generează o întrerupere la fiecare 54,9256 ms, adică de 18,2 ori pe secundă);
- INT 09H = întrerupere dedicată tastaturii, generată la apăsarea şi eliberarea unei taste;
- INT 0BH, INT 0CH = controlează dispozitivele seriale, adică porturile COM1 şi COM2;
- INT 0DH, INT 0FH = controlează dispozitivele paralele, adică porturile LPT1 şi LPT2;
- INT 0EH = controlează operarea dischetei;
- INT 10H = utilizare terminal grafic;
- INT 11H = determinare configurație sistem şi returnează valoarea la locația BIOS 40:10H, în AX;
- INT 12H = determinare dimensiune memorie RAM (returnează în AX dimensiunea memoriei de bază, exprimată în Ko);
- INT 13H = utilizare disc, adică operațiile de intrare/ieşire, formatare etc. (numărul funcției solicitate se transmite în AH);
- INT 14H = utilizare interfață serială prin portul de comunicație serială RS232;
- INT 15H = servicii de sistem determinate de valoarea transmisă în AH, cum ar fi:
  - POST (power-on self-testing, AH=21H), adică testările executate de sistem la pornire;
  - citire stare sistem (AH=43H);
  - suport joystick (AH=84H);
  - determinarea dimensiunii memoriei extinse (AH=88H);
  - comutare procesor în modul protejat (AH=89H);
  - interfața cu mouse-ul (C2H).

După această operație trebui reactivate întreruperile, întrucât revenirea se produce fără a le reactiva.

■ INT 16H = utilizare tastatură;
■ INT 17H = utilizare imprimantă;
■ INT 18H = intrare de bază în ROM; apelată de BIOS dacă sistemul este pornit fără un disc care să conțină programele sistemului de operare DOS;
■ INT 19H = încărcătorul sistemului de operare de pe disc; dacă este disponibilă o dischetă sistem, cu sistemul de operare DOS, va citi pista 0, sectorul 1 în locația de încărcare în memorie la adresa 7C00H şi transferă controlul la această locație; dacă nu există un disc sistem, se transferă controlul la punctul de intrare din ROM, via INT 18H;
■ INT 1AH = utilizare ceas de timp real, adică se poate citi sau inițializa ora în funcție de codul transmis în AH;
■ INT 1BH = permite controlul asupra tastei Break; când se tastează CTRL+Break, ROM BIOS transferă controlul la adresa sa de întrerupere, unde este inițializat un indicator.

Utilizarea terminalului grafic, INT 10H

Sistemul de afişare are două componente de bază: adaptorul video şi monitorul (ecranul monocrom sau color, pe care se afişează text sau imagini grafice). Indiferent de tipul monitorului, acesta permite afişarea unei matrice de puncte pe ecran (pixeli). Caracterele alfanumerice sunt de fapt afişate tot ca o matrice de pixeli, trimisă monitorului de către adaptorul video. Capacitatea unor monitoare de a afişa atât text, cât şi imagini grafice, depinde numai de adaptorul video, nu şi de monitor.

Informația care se afişează pe ecran (text şi imagini grafice) la un moment dat este păstrată în memoria adaptorului video, numită memoria ecran. Această memorie poate fi accesibilă fie utilizând funcții BIOS (INT 10h), fie direct prin program (prin instrucțiuni de transfer), pentru a înscrie informația ce dorim să se afişeze.

Pentru a obține acces direct la această memorie, trebuie să cunoaştem modul de organizare a informației, precum şi spațiul de adrese în care este plasată memoria ecran. Deşi fizic memoria ecran este amplasată pe placa adaptorului video, ea este accesibilă în spațiul de adrese al procesorului. Capacitatea memoriei ecran depinde de tipul adaptorului video, variind intre 64Ko si 512Ko. Modul de organizare al informației în memoria video este determinat de modul de lucru: text sau grafic.

Terminalul video lucrează într-unul din modurile de adresare a I/O: maparea I/O peste memoria calculatorului. Un text ecran de 80×25 caractere este de fapt un vector bidimensional de cuvinte, fiecare cuvânt din acest vector corespunzând unui caracter de pe ecran. Acest vector este memorat imediat după spațiul de memorie de 640K. Dacă se memorează date în acest spațiu utilizând instrucțiunile obişnuite (de tipul mov) se vor afecta caracterele ce apar pe ecran. Adresa memoriei video este B800:0000H.

Un exemplu de adaptor larg răspândit este VGA (Video Graphics Array), succesor al adaptorului EGA (Enhanced Graphics Adapter). Pentru modul grafic, caracteristice sunt modurile: 640 x 480 pixeli şi 16 culori simultan pe ecran (din 256 culori posibile), respectiv 320 x 200 pixeli şi 256 culori simultan pe ecran.

Programele care obțin acces direct la memoria ecran trebuie sa țină cont de tipul adaptorului, deoarece în modul de lucru grafic organizarea informației în memoria ecran şi spațiul de adrese al acesteia diferă de la un tip de adaptor la altul. În modul de lucru text însă, informația este organizată la fel. Accesul la memoria ecran se poate face fie prin intermediul rutinelor din BIOS, care oferă însă avantajul portabilității programelor în detrimentul vitezei de

execuție, fie direct din program, ca la orice locație de memorie, în aplicații care necesită viteze ridicate (de exemplu animație). Pentru a stabili tipul adaptorului video din calculator, se poate utiliza funcția BIOS 1Bh, INT 10h, care întoarce în registrul AL o valoare corespunzătoare (3-adaptor EGA, 4-adaptor VGA).

Terminalul grafic poate lucra în două moduri: alfanumeric (text) şi grafic.

Modurile de lucru alfanumerice sunt caracterizate de numărul de linii caracter (25), numărul de coloane caracter (40 sau 80), atribute de culoare şi numărul de pagini. Fiecare caracter este descris printr-un şablon de puncte. Şablonul cu descrierea formei caracterului se află într-o tabelă din memorie. Pentru acest mod de lucru, un caracter este specificat prin codul ASCII şi atributul de culoare, care conține trei elemente (culoare fond – ultimii 3 biți, intensitate high – 1 bit, culoare caracter – următorii 3 biți, afişare continuă sau intermitentă – 1 bit). Textul poate fi memorat în memoria video în pagini.

O pagină memorează un întreg ecran de text (date) şi este numerotată începând de la 0, pentru modul normal cu 80 de coloane. În acest, mod pagina 0 este implicită şi începe în zona de memorie video de la adresa B800, pagina 1 începe la B900, pagina 2 la BA00, iar pagina 3 la BB00. Se poate afişa o singură pagină pe ecran la un moment dat. Întrucât fiecare caracter de afişat necesită doi octeți de memorie (caracterul, octetul mai puțin semnificativ şi atributul, octetul mai semnificativ) o pagină întreagă de caractere, cu 80 de coloane şi 25 de linii, va necesita 80×25×2=4000 octeți, 2000 caractere şi 2000 fiind ocupați de atributele acestora.

Memoria video alocată efectiv pentru fiecare pagină de memorie este de 4K, adică 4096 octeți, astfel încât ultimii 96 de octeți de după fiecare pagină ecran nu sunt folosiți. Adaptoarele pentru monitoarele color actuale furnizează 32K pentru afişări de text şi permite alegerea uneia dintre cele opt afişări diferite posibile. Fiecare afişare începe la o adresă multiplu de 4K, de la adresele B800:0H, B800:1000H, B800:2000H,..., B800 :7000H. Cele mai multe adaptoare color moderne oferă memorie de la adresa A000:0H până la B000:FFFFH (şi chiar mai mult), dar pentru modul text se utilizează doar cei 32K de la B800:0H până la B800:7FFFH.

Memoria video poate fi adresată exact ca şi memoria RAM. În această memorie se pot memora totuşi şi variabile program sau chiar cod, dar nu este recomandat. Chiar dacă memorarea se realizează într-o pagină ce nu este activă, procedeul este în continuare nerecomandat, întrucât accesul la adaptorul video este foarte lent (memoria principală este de două până la de zece ori mai rapidă).

Pentru un adaptor color, formatul octetului ce reprezintă atributele de afişare este următorul:

- biții 0-3 culoare caracter: 0000-negru, 0001-albastru, 0010-verde, 0011-azuriu (cyan), 0100-roşu, 0101-magenta, 0110-maro, 0111-gri deschis, 1000-gri închis, 1001-albastru deschis, 1010-verde deschis, 1011-azuriu deschis, 1100-roşu deschis, 1101-magenta deschis, 1110-galben, 1111-alb;
- biții 4-6 culoare fundal: culorile corespunzătoare combinațiilor de la 000 la 111 sunt aceleaşi ca şi culorile anterioare din intervalul 000-0111;
- bitul 7: pâlpâire (blinking), dacă este 1, altfel caracterul este afişat normal.

Modul de lucru grafic se caracterizează prin numărul de linii şi coloane (de puncte de pe ecran – pixeli), precum şi prin culoare. Întreruperile manipulează ecranul transferând datele direct în memoria video. Datele se pot transfera direct în memoria video, dar întrucât nu întotdeauna adresele sunt aceleaşi pe toate sistemele, se recomandă utilizarea operațiilor INT 10h şi INT 21h, care cunosc locația memoriei video.

Pentru utilizarea terminalului grafic se pot utiliza următoarele funcții (INT 10h), selectate de conținutul registrului AH, astfel:

Funcția 0 – selecție mod de lucru:

Registrul AL va specifica modul de lucru selectat:

0 = alfanumeric 40 x 25, alb/negru;
1 = alfanumeric 40 x 25, color;
2 = alfanumeric 80 x 25, alb/negru;
3 = alfanumeric 80 x 25, color;
4 = grafic 320 x 200, color;
5 = grafic 320 x 200, alb/negru;
6 = grafic 640 x 200, alb/negru, etc.;
17= grafic, 640 x 480 pixeli, alb/negru (VGA);
18= grafic, 640 x 480 pixeli, 16 culori (VGA);
19= grafic, 320 x 200 pixeli, 256k culori.

Funcția 1 – selecție formă şi dimensiune cursor:

Cursorul nu face parte din setul de caractere ASCII şi există numai în modul text. Forma normală a cursorului este similară caracterului subliniere (underline), dar poate fi extins pe verticală, astfel:

(CH) = numărul liniei inferioare (minim 0);
(CL) = numărul liniei superioare (maxim 7 sau 14 pentru VGA).

Funcția 2 – poziționare cursor pe ecran:

(DH) = numărul liniei;
(DL) = numărul coloanei;
(BH) = pagina.

Funcția 3 – citire coordonate şi formă cursor:

(BH) = numărul paginii pentru care se citeşte poziția.
rezultatul va fi în registrele:
(DH,DL) = linia, coloana;
(CH,CL) = forma cursorului.

Funcția 4 – citire poziție indicator optic:

(AH) = 1; dacă este 0, nu s-a cerut poziția;
(DH,DL) = poziția linie/coloană caracter;
(CH,BX) = poziția linie/coloană, puncte pe ecran (pixeli).

Funcția 5 – selecție pagină activă pe ecran:

(AL)= numărul paginii de activat (numai în mod alfanumeric);

Funcția 6 – defilarea în sus a imaginii (scroll up):

Dacă un program afişează text pe ecran, după ce scrie ultima linie va continua să scrie următoarele linii de text de la începutul ecranului (prima linie). Soluția este de a deplasa liniile scrise în sus şi a insera linii albe (vide) în partea de jos a ecranului. Funcția aceasta permite

ştergerea (defilarea) ecranului sau numai a unei părți a acestuia, definind o fereastră pentru care se pot stabili atributele de afişare (culoare fundal şi cea pentru afişare caractere).

(AL) = numărul de linii de deplasat; 0 pentru întreg ecranul sau fereastra specificată;

(CH,CL) = limite fereastră, colțul din stânga sus, în care se face deplasarea (linie/coloană);

(DH,DL) = limite fereastră, dreapta jos;

(BH) = atribut de culoare pentru liniile introduse.

Iată secvența pentru defilarea cu o singură linie:

```
mov   ax, 0601h        ; defilare cu o linie
mov   bh, 71h          ; atribut fundal (alb-7), culoare caracter (bleu-1)
mov   cx, 00h          ; de la 0:0
mov   dx, 184fh        ; tot ecranul 24:79
int   10h
```

Pentru ca secvența anterioară să şteargă întreg ecranul, este suficient să încărcați în AX valoarea 0600h (în loc de 0601h)

Registrele CX şi DX permit definirea doar a unei ferestre (porțiune rectangulară a ecranului) pentru defilare. În acest caz, trebuie coordonată valoarea din AL cu distanța din CX:DX. De exemplu, vom defini o fereastră delimitată de colțul din stânga sus 12:25 şi cel din dreapta jos 18:54, deci o fereastră de 7 linii şi 30 de coloane, care va fi ştearsă.

```
mov   ax, 0607h        ; defilare cu 7 linii
mov   bh, 30h          ; atribut fundal, culoare caracter
mov   cx, 0C19h        ; de la 12:25
mov   dx, 1236h        ; pana 18:54
int   10h
```

Deoarece atributul pentru o fereastră rămâne activ până când altă operație îl modifică, se pot defini în acelaşi timp mai multe ferestre cu atribute diferite.

Funcția 7 – defilare în jos a imaginii (scroll down):

Se utilizează aceleaşi registre ca la funcția de deplasare în sus, cu aceeaşi semnificație, a imaginii dintr-o fereastră, conținută în pagina activă, dar de data aceasta deplasarea se face în jos şi liniile vide apar în partea de sus a ferestrei.

Funcția 8 – citire caracter şi atributele sale de pe ecran, din dreptul cursorului:

(BH) = pagina referită.

După apelul funcției, valorile returnate sunt:

(AL) = codul ASCII al caracterului citit;

(AH) = atribut de culoare (pt. alfanumeric).

Funcția 9 – afişare caractere pe ecran, la poziția cursorului:

(BH) = numărul paginii în care se face afişarea;

(AL) = codul ASCII al caracterului de afişat;
(CX) = numărul de repetări ale afişării acestui caracter;
(BL) = atribut de culoare.

Această operație nu avansează cursorul şi nici nu răspunde la caractere de tip semnal sonor (bell), linie nouă (line feed), revenire la început de linie (carriage return) sau tabulare (tab); în schimb, sunt afişate ca şi caractere ASCII. Dacă afişarea depăşeşte linia respectivă, se va continua afişarea pe linia următoare, de la capăt.

Funcția 12 (0CH) – afişare punct pe ecran (pixel):
(DX) = numărul liniei;
(CX) = numărul coloanei;
(AL) = culoare punct (dacă primul bit este 1, se va face un „sau exclusiv“ între noua culoare şi cea curentă).

Modurile de lucru VGA 0DH, 0EH, 0FH şi 10H furnizează 16, 8, 4 şi respectiv 2 pagini de memorie video. Numărul paginii implicite este 0.

Funcția 13 (0DH) – citire culoare punct de pe ecran:
(DX,CX) = numărul liniei/coloanei punctului;
(AL) = codul culorii punctului referit.

Funcția 14 (0EH) – afişare caracter şi actualizare poziție cursor:
(AL) = codul ASCII al caracterului de afişat;
(BL) = culoare caracter, pentru modul grafic;
(BH) = numărul paginii în care se face afişarea.

Această funcție permite utilizarea monitorului ca terminal pentru afişare. După afişarea caracterului, cursorul este avansat pe poziția următoare, pe aceeaşi linie sau la început de linie sau de pagină; caracterele speciale (“bell“=7, „backspace“=8, „line feed“=10, „carriage return“=13) sunt interpretate ca şi comenzi pentru formatarea ecranului.

Funcția 15 (0FH) – citire caracteristici mod de lucru:
După apelul funcției, aceasta returnează următoarele valori:
(AL) = numărul modului de lucru;
(AH) = numărul de coloane caracter (caractere pe linie: 20, 40 sau 80);
(BH) = numărul paginii curente.

Utilizarea modului grafic

În modul grafic, pe ecran se afişează imagini grafice alcătuite din puncte (pixeli). Numărul de biți prin care se reprezintă un pixel în memoria ecran depinde de numărul de culori ce pot fi afişate pe ecran. Astfel, pentru un monitor monocrom, este suficient un singur bit, cu semnificația bit aprins(1), respectiv bit stins (0). În general, pentru a afişa 2^n culori sunt necesari n biți/pixel. Ecranul este considerat ca fiind împărțit într-o matrice de puncte, fiecare având culoare proprie. Dimensiunea matricei de pixeli determină rezoluția care se defineşte ca fiind:

rezoluția = număr_pixeli_pe_orizontală × număr_pixeli_pe_verticală

În funcție de rezoluție şi de numărul de culori ce se pot afişa simultan pe ecran, în memoria video se pot păstra una sau mai multe imagini ecran, numite pagini video. Conceptul de pagină video (ecran) este utilizat şi în mod text. Folosirea paginilor video se recomandă în aplicații de animație pe calculator sau în afişarea unor imagini complexe, care necesită un timp mare de generare a imaginii.

Modul grafic utilizează la afişare pixeli pentru a genera (desena) pe ecran, în diverse culori. De exemplu, funcția 0, INT 10H, cu o valoare în AL mai mare decât 4 va defini un mod grafic; pentru AL = 10H este modul grafic, color, cu rezoluția 640×350. Fiecare octet din memorie reprezintă 4 pixeli (câte doi biți pentru fiecare pixel), numerotați de la 0 la 3. Sunt disponibile patru culori, în orice moment, numerotate de la 0 la 3. Limitarea numărului de culori este dată de numărul de biți utilizați (2, deci în total 4 combinații). Dar pixelul 0 poate fi ales din cele 16 culori pentru fundal (background). Ceilalți trei pixeli pot fi aleşi din două palete de culori, de câte trei culori. Funcția 0BH, INT 10H poate fi utilizată pentru a selecta o paletă de culori şi fundalul. O paletă conține patru culori: trei culori (ex. gri, roşu şi maro) plus culoarea de fundal definită.

Funcțiile modului grafic, dintre cele definite anterior, sunt următoarele: 04H – citire poziție indicator optic, 08H – citire caracter şi atributele sale de pe ecran, din poziția curentă a cursorului, 09H – afişare atribute sau caracter pe ecran, la poziția cursorului, 0AH – înlocuire caractere cu păstrarea culorii, 0BH – fixare caracteristici de culoare, 0CH – afişare punct pe ecran (pixel), 0DH – citire culoare punct de pe ecran.

Vom exemplifica utilizarea unora dintre aceste funcții prin următorul exemplu care trasează pe un anumit fundal linii, în modul grafic, între anumite coordonate de pe ecran, după care se restabileşte modul de lucru inițial al terminalului grafic.

```
; dispvido.asm – program care afiseaza direct in memoria video
; programul traseaza linii prin pixeli de diverse culori
.model small
.stack
.code
        org 100h
start   proc    near
        mov     ah, 0fh         ; citire si salvare mod video initial
        int     10h
        push ax
        call smod               ; setare mod grafic
        call afisg              ; afisare in modul grafic
        call citasta            ; terminare mod grafic prin apasarea unei taste
        pop     ax              ; refacerea modului grafic initial
                                ; mov ah,0
        int     10h
        mov     ax, 4c00h       ; revenire DOS
        int     21h
start   endp
```

```
smod    proc    near        ; setarea modului grafic VGA 640 x 350
        mov     ah, 0
        mov     al, 10h
        int     10h
        mov     ah, 0bh     ; caracteristici de culori
        mov     bh, 0       ; culoare fundal:
        mov     bl, 01h     ; albastru
        int     10h
        ret
smod    endp

afisg   proc    near        ; afisare in modul grafic
        mov     bx, 0       ; pagina initiala 0
        mov     dx, 70      ; linia pe care se incepe afisarea
        mov     cx, 140     ; si coloana de inceput
iar:
        mov     ah, 0ch     ; functia de afisare pixel
        mov     al, bl      ; culoarea de afisare
        int     10h         ; registrele BX, CX si DX nu sunt modificate
        inc     cx          ; incrementare coloana de afisare
        cmp     cx, 500     ; limita coloanei
        jne     iar
        mov     cx, 140     ; daca s-a ajuns la limita, se reia
        inc     bl          ; modificare/incrementare culoare
        inc     dx          ; incrementare linie
        cmp     dx, 280     ; daca nu s-au parcurs toate liniile
        jne     iar         ; se reia afisarea
        ret
afisg   endp

    citasta     proc    near
        mov     ah, 10h     ; citirea unei taste
        int     16h
    citasta     endp
    end start
```

În modul grafic, organizarea informației în memoria ecran depinde de tipul adaptorului video şi de modul de lucru grafic selectat. Astfel, pentru adaptorul VGA, modul 18 (640 x 480 pixeli, 16 culori simultan), memoria ecran este organizată în 4 plane. În fiecare plan, un bit corespunde unui pixel ecran. Pentru a obține valoarea unui pixel se adresează simultan cele 4 plane, cu aceeaşi adresă. Într-un plan de memorie, informația este organizată astfel: biții din primul octet corespund primilor 8 pixeli din linia 0, cel de-al doilea octet corespunde următorilor 8 pixeli etc.

Valoarea unui pixel, de 4 biți din memoria ecran, nu reprezintă culoarea pixelului, ci un indice într-o tabelă de culori, numită paletă. Numărul de culori din paletă determină numărul de culori ce pot fi afişate simultan pe ecran. Valoarea unui pixel fiind reprezentată pe 4 biți, paleta

are 24 (16) culori. O culoare se reprezintă prin diferite valori ale componentelor de bază RGB (Red, Green, Blue).

Adaptorul VGA păstrează, pentru compatibilitate cu EGA, conceptul de paletă cu 16 intrări, 6 biți/intrare. Dar intrarea în paletă nu mai reprezintă o culoare, ci un registru de culoare. Adaptorul VGA dispune în total de 256 de registre de culoare. Un registru de culoare conține o culoare reprezentată pe 18 biți – câte 6 biți pentru fiecare componentă RGB. Culoarea reprezentându-se pe 18 biți, adaptorul VGA poate genera maxim 218 (256k) culori distincte.

În modul de lucru 19, memoria grafică nu mai este organizată în plane. Unui pixel îi corespunde un octet în memoria ecran. Valoarea pixelului reprezintă un registru de culoare din cele 256 ale adaptorului. În memoria ecran, începând de la adresa segment 0A000h, primii 320 octeți corespund liniei 0 de pixeli, următorii 320 octeți corespund liniei 1 etc.

Pentru adaptorul video VGA în modul grafic 19 (rezoluție 320x200, 256 culori simultan pe ecran), accesul la memoria ecran este mult mai simplu decât în modul 18. Offset-ul octetului ce conține valoarea unui pixel (x,y) este:

offset = 80 x y + x

Caracteristic acestui mod de lucru este numărul mare de culori ce pot fi afişate simultan pe ecran. Funcția INT 10h oferă un set de subfuncții referitoare la paleta şi la registrele de culoare. Citirea, respectiv scrierea registrelor de culoare se poate realiza cu funcțiile:

- 10h modifică (scrie) un registru culoare; date de intrare:
  - BX = index registru culoare;
  - DH, CH, CL = valori RGB (6 biți din fiecare registru, cei mai puțin semnificativi).
- 12h modifică (scrie) un set (bloc) de registre de culoare; date de intrare:
  - BX = indexul primului registru culoare din set;
  - CX = contor registre culoare din set
  - ES:DX = adresa tabelei de culori, o culoare fiind reprezentată pe 3 octeți (corespunzător componentelor RGB).
- 15h citeşte un registru culoare; date de intrare:
  - BX = index registru culoare, date de ieşire:
  - DH, CH, CL = valorile RGB (6 biți din fiecare registru, cel mai puțin semnificativi).
- 17h citeşte un set (bloc) de registre de culoare; date de intrare:
  - BX = indexul primului registru culoare din set,
  - CX = contor registre culoare din set, ES:DX = adresa tabelei de culori, o culoare fiind reprezentată pe 3 octeți (corespunzător componentelor RGB), date de ieşire: valorile din registrele culoare din set, pe care le înscrie în tabela de culori.

Utilizarea discului, INT 13H

Pe lângă serviciile DOS oferite de funcția INT 21H, se mai poate utiliza şi funcția BIOS INT 13H pentru a opera direct cu discul, cu toate că funcțiile BIOS nu oferă utilizarea automată a directoarelor sau blocarea, respectiv deblocarea înregistrărilor. Funcția INT 13H tratează datele ca dimensiunea unui sector şi manipulează adresarea discului folosind numerele efective ale pistei şi sectorului. Operațiile INT 13H implică inițializarea, citirea, scrierea, verificarea şi formatarea discului. Ele sunt destinate programatorilor cu experiență care trebuie să fie atenți în utilizarea lor, precum şi la faptul că pot fi diferențe între diferite versiuni de BIOS.

Operațiile furnizate de INT 13H, selectate de conținutul lui AH, sunt următoarele:

00H	Inițializare unitate /disc sistem	0CH	Căutare cilindru
01H	Citire stare unitate /disc	0DH	Inițializare disc alternativ
02H	Citire sectoare	0EH	Citire buffer sector
03H	Scrie sectoare	0FH	Scriere buffer sector
04H	Verifica sectoare	15H	Determinare tip disc
05H	Formatare piste	16H	Schimbare stării dischetei
08H	Citire parametrii unitate disc	17H	Inițializare tip dischetă
09H	Inițializare unitate	18H	Inițializare tip media pt. format
0AH	Citire buffer sector extins	19H	Parcare capete disc
0BH	Scriere buffer sector extins		

Majoritatea operațiilor INT 13H pun pe 0 sau pe 1 indicatorul CF, după cum operația respectivă a reuşit sau nu, şi returnează un cod de stare în registrul AH. BIOS păstrează informația în zona sa de date, pentru fiecare unitate de disc. Octetul de stare este prezentat în continuare şi se poate găsi în zona de date BIOS, la adresa 40:41H pentru zona de date pentru dischetă (Diskette Drive Data Area) şi la adresa 40:74H pentru hard disc (Hard Disk Data Area).

Codul	Starea
00H	Nu este eroare
01H	Comandă eronată, nerecunoscută de controller
02H	Adresa sau discul nu s-au găsit
03H	Încercare de scriere pe un disc protejat
04H	Pistă / sector invalid
05H	Operație de inițializare eşuată
06H	Dischetă scoasă/ înlocuită de la ultimul acces
07H	Parametrii disc eronați
08H	Depăşire DMA
09H	Încercare de scriere/ citire DMA peste limita de 64K
10H	Eroare CRC la citire (indică date corupte)
20H	Eroare hardware, controller defect
40H	Eroare hardware, operație căutare eşuată
80H	Dispozitivul nu răspunde (uşă unitate deschisă, nu este dischetă, sau pentru hard disk depăşire timp răspuns)
AAH	Unitatea nu este pregătită
BBH	Eroare nedefinită
CCH	Eroare la scriere

Dacă o operație cu discul returnează o eroare, acțiunea obişnuită a unui program este să inițializeze discul (funcția 00H) şi să reia operația respectivă de încă trei ori. Dacă eroarea persistă, programul poate afişa un mesaj care să permită utilizatorului să schimbe discheta, dacă aceasta este cauza.

Operațiile de bază permise de funcția DOS INT 13H sunt următoarele:

Funcția 00H – inițializare disc sistem:

- (DL) = unitatea de disc: 0-discul A, 80H-primul hard disc, 80H-al doilea hard etc.

Se utilizează după ce o operație anterioară a semnalat o eroare severă; operația realizează o reinițializare hard a controllerului de disc sau hard disc, adică la următorul acces se va inițializa operația de la cilindrul 0. Dacă operația este reuşită CF=0, în caz contrar va fi 1, iar AH va conține codul de stare.

Funcția 01H – citire stare disc:

- (DL) = unitatea de disc: 0-discul A, 80H-primul hard disc, 80H-al doilea hard etc.

Operația returnează în AL codul de stare pentru ultima operație cu discul (returnat în AH). Operația validă va pune CF=0 şi AH=00H.

Funcția 02H – citire sectoare disc:

Operația permite citirea unui număr specificat de sectoare de pe aceeaşi pistă direct în memorie. Pentru aceasta, trebuie inițializate următoarele registre:

- (AL) = numărul de sectoare (până la numărul maxim de sectoare de pe o pistă);
- (CH) = numărul de cilindru/pistă (încep de la 0);
- (CL) = biții 7-6 conțin numărul cilindru/pistă (primii doi biți ai numărului, cei mai semnificativi), iar biții 5-0 conțin numărul sectorului de start (încep de la 1);
- (DH) = numărul cap/față (0 sau 1 pentru dischetă);
- (DL) = numărul unității (precizat şi anterior, 0-A, 80H-primul hard disc etc.)
- (ES:BX) = adresa unui buffer în zona de date, care trebuie să fie suficient de mare pentru toate sectoarele de citit.

Dacă operația a reuşit, CF=0 şi se returnează în AL numărul efectiv de sectoare citite, iar celelalte registre îşi păstrează valorile. Dacă apare o eroare, CF=1, iar AH va conține codul de stare.

Funcția 03H – scriere sectoare disc:

Operația scrie conținutul unei zone specificate de memorie de 512 octeți sau multiplu de 512 octeți în sectoarele specificate. Registrele utilizate pentru a specifica aceşti parametri sunt aceleaşi ca la funcția 02H, iar starea de reuşită a operației este semnalată asemănător.

Funcția 04H – verificare sectoare disc:

Operația permite verificarea unui număr specificat de sectoare, în sensul că se pot citi datele şi se verifică CRC (Cyclical Redundancy Check). Când se scrie pe disc, într-un sector controllerul de disc calculează şi scrie o sumă de control (CRC) imediat după sectorul respectiv. Funcția 04H citeşte sectorul, recalculează suma de control şi o compară cu suma scrisă. Verificarea constă din recalcularea acestei sume şi nu din compararea octeților din sector cu aceia din memorie. Această operație poate fi utilizată după o scriere (funcția 03H) pentru a fi siguri de informația scrisă, deşi necesită un timp suplimentar de procesare.

Parametrii se încarcă în registre exact ca la funcția 02H, dar nu este necesară adresa din ES:BX, întrucât nu se verifică datele scrise în memorie. Starea de reuşită este semnalată la fel ca la funcția 02H.

Funcția 05H – formatare piste:

Operația permite formatarea pistelor, pentru a permite operațiilor de citire/scriere să localizeze şi să proceseze un anumit sector. Înainte de a folosi această operație, trebuie utilizată funcția 17H pentru a stabili tipul dischetei şi funcția 18H pentru a stabili tipul mediului. În vederea formatării, trebuie inițializate următoarele registre:

- (AL) = numărul de sectoare de formatat;
- (CH) = numărul de cilindru/pistă (încep de la 0);
- (CL) = biții 7-6 conțin numărul cilindru/pistă (primii doi biți ai numărului, cei mai semnificativi), iar biții 5-0 conțin numărul sectorului de start (încep de la 1);
- (DH) = număr cap/față (0 sau 1 pentru dischetă);
- (DL) = numărul unității (0-A, 80H-primul hard disc etc.);
- (ES:BX) = adresa către un grup de câmpuri de adresă pentru o pistă. Pentru fiecare sector de dischetă pentru o pistă, trebuie să se găsească o intrare de 4 octeți de forma T/H/S/B, unde semnificația acestor octeți este următoarea:
  - T (0) – număr cilindru/pistă;
  - H (1) – număr cap/față;
  - S (2) – număr sector;
  - B (3) – număr de octeți/sector (00H=128, 01H=256, .., 03H=1024H)

De exemplu, pentru a formata pista 03, capul 00, 512 octeți/sector, prima intrare pentru pistă este 03000102, urmată de o intrare pentru fiecare sector rămas.

Dacă operația a reuşit, CF=0, altfel CF=1, iar AH va conține codul de stare.

Utilizarea tastaturii, INT 16H

Tastatura conține un controller (8042) care scanează constant comutatoarele tastaturii (comandate de taste) pentru a determina dacă este vreo tastă apăsată. Acest proces se desfăşoară în paralel cu activitățile normale ale calculatorului, astfel încât să nu se piardă nici o apăsare de tastă. Codul de scanare furnizat de tastatură este disponibil la portul 60H.

Pentru tastatură există, de fapt, două proceduri: una pe nivelul 09H de întreruperi, care tratează semnalele venite de la tastatură, şi una pe nivelul 16H, care poate fi folosită de utilizator pentru obținerea unui caracter de la tastatură. Prima procedură depune caracterele transmise de tastatură într-un buffer (de 16 caractere), în timp ce a doua procedură preia caractere din acest buffer. Bufferul este organizat circular, şi conține pentru fiecare caracter transmis o pereche de octeți (codul ASCII şi codul de scanare – numărul tastei care l-a furnizat); la depăşirea capacității bufferului se emite un semnal sonor.

Multe din codurile ASCII sunt caractere tipăribile (litere, cifre, semne speciale etc.), dar din cele 256 de coduri, multe nu sunt reprezentate pe tastatură. Pentru a introduce astfel de coduri sau pentru a introduce un cod oarecare, se ține apăsată tasta <Alt> şi se tastează codul respectiv ca o valoare zecimală de la tastele numerice (în intervalul 0-255). Valoarea astfel tastată este memorată pe doi octeți în bufferul tastaturii, primul este caracterul ASCII, iar cel de-al doilea este 0.

Procedura pentru tratarea întreruperilor de la tastatură realizează interpretarea şi execuția unor comenzi corespunzătoare şi a unor combinații de taste speciale:

CTRL/ALT/DEL = reinițializare sistem;

SHIFT/PRTSC = copiere la imprimantă a caracterelor de pe ecran;

CTRL/NUM LOCK = oprirea execuției unui program până la apăsarea altei taste;

CTRL/SCROLL LOCK = încheie execuția unui program; aici utilizatorul îşi poate defini propria sa rutină de tratare a întreruperii (1BH).

Tastatura furnizează trei tipuri de taste funcționale, de bază:

1. Caracterele standard (litere, cifre, caractere speciale care se află pe tastatură).
2. Tastele pentru funcțiile extinse, care constau din:
   - taste funcționale, precum F1 sau Shift + F1.
   - tastele numerice din partea dreaptă a tastaturii, cu tasta NumLock inactivă: <Home>, <End>, tastele direcționale <□, □, □, □>, <Del>, <Ins>, <PgUp>, <PgDn>, precum şi duplicatele lor de pe tastatură.
   - <Alt> + literă şi <Alt> + taste funcționale
3. Taste de control: <Alt>, <Ctrl>, <Shift>, care lucrează în combinație cu alte taste. BIOS-ul le tratează în mod diferit față de celelalte prin actualizarea stării lor curente (apăsată sau nu) în octeții de stare ai tastelor de tip <Shift> în zona de date BIOS.

Primele PC-uri aveau 83 de taste. Dintre acestea, cele numerice din partea dreaptă pot realiza două tipuri de funcțiuni. Ele pot reprezenta, pe lângă numere, şi tastele <Home>, <End>, tastele direcționale <←, ↑, →, ↓>, <Del>, <Ins>, <PgUp>, <PgDn>, selecția fiind realizată prin tasta <NumLock>. Ulterior, la tastaturile următoare, această problemă a fost rezolvată prin proiectarea tastaturilor cu 101 şi apoi cu 104 taste pentru Windows. Dintre tastele noi, doar F11 şi F12 sunt funcții noi, restul menținând funcțiile tastelor de la tastatura originală. Bineînțeles că şi BIOS-ul a fost modificat, pentru a putea interpreta şi aceste noi taste.

Starea tastelor de tip <Shift>

Zona de date BIOS, aflată începând de la adresa de segment 40H, conține şi un prim octet cu starea tastelor de tip <Shift> (la adresa 40H: 17H) cu următoarea configurație:

7	6	5	4	3	2	1	0
Insert	Caps Lock	Num Lock	Scroll Lock	Alt	Ctrl	Left Shift	Right Shift

Pentru citirea lor se foloseşte funcția 02H, int 16H; o tastă activă (1) înseamnă că utilizatorul ține apăsată acea tastă. Acesta este octetul pentru tastatura cu 83 de taste.

Tastaturile îmbunătățite (cu 101 sau 104 taste), care au duplicate şi pentru tastele <Ctrl> şi <Alt> necesită informație suplimentară pentru a le testa. Din acest motiv se utilizează un al doilea octet de stare la adresa 40:18H în care un bit 1 înseamnă că sunt active următoarele taste:

7	6	5	4	3	2	1	0
Insert	Caps Lock	Num Lock	Scroll Lock	Ctrl/ Num	SysReq	Left Alt	Left Ctrl

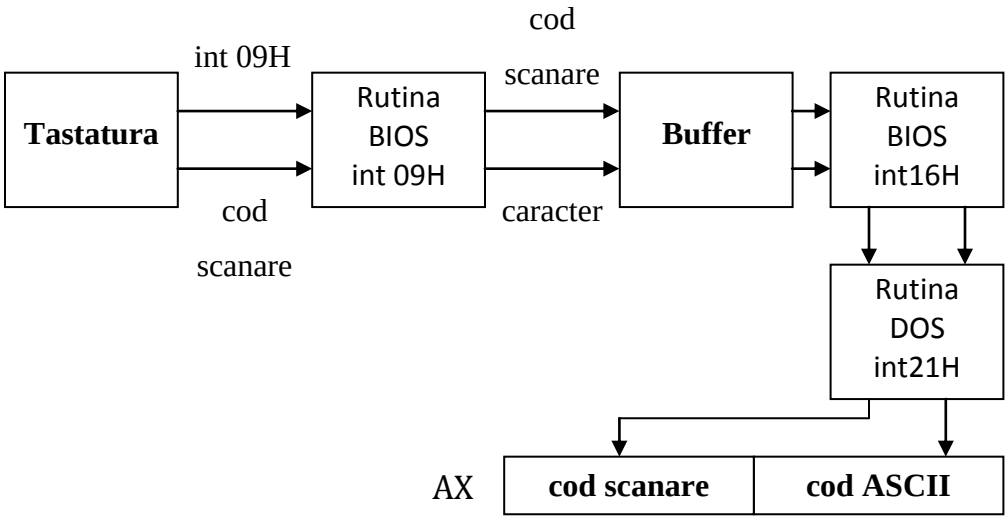
Un alt octet de stare se află la adresa 40:96H, care conține şi alte informații, din care bitul 4 specifică dacă este instalată o tastatură îmbunătățită.

Bufferul tastaturii

Transferul datelor între tastatură şi programele utilizator se face printr-o zonă de memorie tampon, denumită buffer. Zona de date BIOS conține la adresa 40:1EH o zonă tampon (bufferul) pentru datele introduse de la tastatură. Aceasta permite introducerea până la 15 caractere, înainte ca

programul să solicite intrarea. Pentru fiecare tastă este asociat un cod unic de scanare, transmis la apăsarea tastei (denumit cod „make“); la eliberarea tastei se mai transmite un cod obținut prin adunarea valorii 80H la codul de scanare, denumit cod „break“. La apăsarea unei taste, se solicită în mod automat întreruperea BIOS INT 09H. Aceasta ia codul scan de la tastatură, îl convertește în cod ASCII şi-l depune în bufferul tastaturii. Programul utilizator pentru citire din buffer va folosi INT 16H.

Când se apasă o tastă, procesorul tastaturii generează codul scan şi o cerere de întrerupere, INT 09H, care lansează rutina asociată întreruperii din BIOS. Aceasta citeşte de la portul de intrare 96 (in al, 60H) codul de scanare, pe care-l compară cu intrarea dintr-o tabelă de coduri de scanare pentru a determina codul ASCII ataşat caracterului, dacă acesta există. Rutina va furniza cele două coduri pe doi octeți (în registrul AX) în buffer-ul tastaturii. Procedeul acesta este prezentat în figura următoare.



Valorile returnate în registrul AX de întreruperi (BIOS sau DOS) se află în ordine inversă celei din bufferul tastaturii (în AH se găseşte codul de scanare şi în AL codul ASCII pentru tastele ce au asociate coduri, respectiv 0 pentru cele ce nu au un astfel de cod).

Întreruperea INT 09H manipulează octeții de stare ai tastaturii (de la adresele 40:17H, 40:18H şi 40:96H) pentru <Shift>, <Alt> şi <Ctrl>, dar nu generează nici un caracter în buffer la apăsarea acestor taste, ci doar actualizează biții corespunzători din octeții de stare.

În cazul în care codul de scanare este generat de tastatură la apăsarea unei taste, la eliberarea ei, într-un interval de 0,5 sec. se va genera un al doilea cod de scanare (care este primul cod +80H, adică se pune primul bit pe 1). Cel de-al doilea cod indică rutinei de întrerupere că tasta a fost eliberată. Dacă se ține apăsată tasta mai mult de 0,5 s, atunci se repetă automat operația (tasta).

Bufferul tastaturii păstrează doi pointeri: unul la începutul (capul) bufferului pentru a specifica întreruperii INT 16H de unde se va citi (extrage) următorul caracter şi unul la sfârşitul (coada) buffer-ului, pentru a preciza întreruperii 09H unde se memorează următorul caracter tastat. Următoarele adrese descriu conținutul buffer-ului:

- 40:1AH – adresa (indicatorul) de început buffer, următoarea poziție de unde int 16H citeşte un caracter;
- 40:1CH – adresa de sfârşit buffer, următoarea poziție unde int 09H poate memora caracterul introdus;
- 40:1EH – adresa de început a bufferului însuşi. El conține 32 de octeți (16 caractere), dar

poate depăşi această dimensiune, şi conține caracterele tastate şi codurile lor de scanare. INT 16H va citi câte doi octeți, fiecare caracter şi codul său de scanare, şi îi furnizează programului.

Acest buffer al tastaturii este unul circular, în sensul că indicatorul de sfârşit este avansat de INT 16H. În acest fel, indicatorul de început urmăreşte continuu indicatorul de sfârşit. Când cei doi indicatori se află pe aceeaşi poziție, coada este goală (int 16H a citit toate caracterele). Când utilizatorul introduce 15 caractere, bufferul este plin şi indicatorul de sfârşit se găseşte imediat în spatele celui de început. În acest moment, int 09H nu mai acceptă tastarea de noi caractere, bufferul acceptă numai 15 caractere, cel mult, deşi poate conține 16.

Structura acestui buffer poate fi descrisă astfel:

```
BIOS_data segment at 40H
                ORG 17H
                stare_KBD     db      ?
                ORG 1AH
                Buf_head      dw      ?       ; pointer de extragere din buffer
                Buf_tail      dw      ?       ; pointer de introducere in buffer
                Buf_KBD       dw 16 dup (?); bufferul tastaturii
BIOS_data ends
```

Următorul program (.com) afişează starea tastelor Shift, Ctrl şi Alt.

```
; tastshft.asm – programul afiseaza daca s-au tinut apasate tastele
; „Shift“, „Ctrl“ si „Alt“
BIOS_date               segment at 40h
        org 17h
stare_KBD               db      ?
BIOS_date               ends

cod             segment                 para
        assume cs: cod, es: BIOS_date
        org     100h
start:  mov ax, BIOS_date
        mov es,ax
citire: mov ah, 10h             ; citire caracter de la tastatura
        int 16h
        cmp al,0dh              ; caracterul de sfarsit este tasta „Enter“
        je gata
        call test_Shift         ; apel functie afisare stare taste „Shift“
        jmp citire
gata:
        mov ax, 4c00h           ; revenire DOS
        int 21h

test_Shift      proc    near
```

```
                push cs                 ; initializarea registrului segment DS
                pop ds                  ; cu adresa segmentului unde sunt mesajele
                mov bl, stare_KBD
                test bl, 01h            ; test apasare tasta „Shift“ dreapta
                jz test_Shift_stg
                lea dx, mes_Shift_dr
                call tip_mes
        test_Shift_stg:
                test bl, 02h            ; test apasare tasta „Shift“ stanga
                jz test_Ctrl
                lea dx, mes_Shift_stg
                call tip_mes
        test_Ctrl:
                test bl, 04h            ; test apasare tasta „Ctrl“
                jz test_Alt
                mov bh, stare_KBD[1]; test „Ctrl“ stanga sau dreapta
                test bh, 01h            ; tasta „Ctrl“ stanga
                jz Ctrl_dr
                lea dx, mes_Ctrl_stg
                call tip_mes
                jmp test_Alt
        Ctrl_dr:
                lea dx, mes_Ctrl_dr
                call tip_mes
        test_Alt:
                test bl, 08h            ; test apasare tasta „Alt“
                jz gata_test
                mov bh, stare_KBD[1 ; test „Alt“ stanga sau dreapta
                test bh, 02h            ; tasta „Alt“ stanga
                jz Alt_dr
                lea dx, mes_Alt_stg
                call tip_mes
                jmp gata_test
        Alt_dr:
                lea dx, mes_Alt_dr
                call tip_mes
        gata_test:
                ret
        test_Shift      endp

tip_mes     proc    near
                mov ah,09h              ; tiparire mesaj de la adresa data in DX
                int 21h
                ret
        tip_mes         endp
```

```
; mesajele de tiparit
mes_Shift_dr        db    'S-a apasat tasta „Shift“ dreapta.',0dh, 0ah, '$'
mes_Shift_stg       db    'S-a apasat tasta „Shift“ stanga.',0dh, 0ah, '$'
mes_Alt_dr          db    'S-a apasat tasta „Alt“ dreapta.',0dh, 0ah, '$'
mes_Alt_stg         db    'S-a apasat tasta „Alt“ stanga.',0dh, 0ah, '$'
mes_Ctrl_dr         db    'S-a apasat tasta „Ctrl“ dreapta.',0dh, 0ah, '$'
mes_Ctrl_stg        db    'S-a apasat tasta „Ctrl“ stanga.',0dh, 0ah, '$'
cod     ends
end
```

Pentru utilizarea tastaturii se pot apela următoarele funcții, selectate de valoarea din registrul AH:

Funcția 0 – citire caracter:

- (AH) = codul de scanare al caracterului;
- (AL) = codul ASCII al caracterului;

Funcția acceptă tastele corespunzătoare tastaturilor cu 83 de taste, dar nu şi tastele corespunzătoare tastaturilor îmbunătățite (<F11> şi <F12>); pentru acestea, se poate utiliza funcția 10H.

Funcția 1 – test caracter disponibil:

Funcția verifică dacă există un caracter disponibil în buffer, dar nu şi pentru tastele corespunzătoare tastaturilor îmbunătățite (<F11> şi <F12>); pentru acestea, se poate utiliza funcția 11H. Caracterul rămâne în buffer până când este citit (cu una din funcțiile 00h sau 10h)

- (ZF) = 0, dacă există caracter disponibil în buffer;
  = 1, dacă nu există caracter în buffer.
- (AX) = cei doi octeți corespunzători caracterului (codul de scanare şi codul ASCII);

Funcția 2 – citire stare taste de tip „shift“:

- (AL) = octetul care codifică starea acestor taste, memorat la adresa 40:17H.

Funcția 3 – inițializarea ratei pentru autorepetare:

Registrul BH conține intervalul de timp de aşteptare înainte de începerea operației de autorepetare a tastei ce este ținută apăsată. Registrul BL va conține rata de repetare, adică numărul de repetări ale tastei respective, într-o secundă.

- (BH) = 0, 1, 2, 3 pentru 1/4, 1/2, 3/4 sau 1 secundă întârziere până la repetare;
- (BL) = 0-1Fh pentru 30 repetări/s până la 2 repetări/s.

Funcția 5 – scriere în bufferul tastaturii:

Funcția permite programelor utilizator să insereze caractere în bufferul tastaturii dacă utilizatorul a apăsat o tastă. Se va încărca codul ASCII în CH şi codul său de scanare în CL. Operația permite introducerea de caractere până la umplerea bufferului.

Funcția 10H – citire caracter (inclusiv tastele duplicate de la tastatura îmbunătățită):

- (AH) = codul de scanare al caracterului;
- (AL) = codul ASCII al caracterului; pentru tastele funcțiilor extinse (<Home>, <F1>) returnează 00H, iar pentru tastele extinse de control (duplicatele de la tastatura îmbunătățită) returnează E0H.

Funcția acceptă toate tastele corespunzătoare atât tastaturilor cu 83 de taste, cât şi tastaturilor îmbunătățite (duplicatele <Home>, <PgUp>, dar şi cele noi <F11> şi <F12>).

Funcția 11H – test caracter disponibil:

Funcția verifică dacă există un caracter disponibil în buffer, dar, spre deosebire de funcția 01H, această funcție verifică pentru toate tastele corespunzătoare oricărui tip de tastatură, adică şi pentru tastaturile îmbunătățite. Utilizarea ei este similară, în rest, cu aceea a funcției 01H.

Funcția 12H – citire stare taste de tip „shift" (pentru tastaturile îmbunătățite):

- (AL) = octetul care codifică starea acestor taste, după cum urmează:

7	6	5	4	3	2	1	0
SysReq	Caps Lock	Num Lock	Scroll Lock	Right Alt	Right Ctrl	Left Alt	Left Ctrl

Utilizarea ceasului de timp real, INT 1AH

Pentru ceasul de timp real, BIOS-ul conține două proceduri: una pentru tratarea întreruperilor provenite de la acesta şi alta disponibilă utilizatorului, pentru controlul ceasului de timp real.

Prima procedură pentru tratarea întreruperilor de la ceasul de timp real este apelată de aproximativ 18,21 ori pe secundă. La fiecare apel, se incrementează un contor format din 4 octeți (de la adresa 40:6CH), ce reprezintă ora sistemului. La fiecare apel al acestei proceduri, după actualizarea contorului se va invoca întreruperea software INT 1CH. Aceasta este procedura pe care o poate defini utilizatorul; dacă el nu a definit-o, aceasta va conține doar instrucțiunea IRET, şi deci nu se va realiza nici o prelucrare.

Pentru utilizarea ceasului de timp real se pot selecta următoarele funcții:

Funcția 0 – citire contor, pentru ceasul de timp real:

- (CX) = partea cea mai semnificativă a contorului;
- (DX) = partea cea mai puțin semnificativă a contorului;
- (AL) = indicator de depăşire a 24 ore (este 0 dacă nu a trecut de ora 24)

Valoarea de 32 de biți din CX:DX reprezintă numărul de perioade de 55 milisecunde care au trecut de la miezul nopții.

Funcția 1 – inițializare contor, pentru ceasul de timp real:

- (CX) = cei mai semnificativi doi octeți pentru contor;
- (DX) = cei mai puțin semnificativi doi octeți pentru contor

Funcția permite inițializarea ceasului de timp real la valoarea conținută în CX:DX, ca număr de perioade de timp de 55 milisecunde scurse de la miezul nopții.

Celelalte funcții, 02H-07H, manipulează ora şi data pentru serviciile de ceas în timp real.

Utilizarea funcțiilor DOS

În timp ce BIOS-ul permite manipularea dispozitivelor la nivel coborât, DOS-ul furnizează o interfață de nivel ridicat pentru multe dispozitive. De exemplu, una dintre rutinele BIOS permite accesul la unitatea de dischetă. Cu această rutină se pot citi sau scrie blocuri pe dischetă. Din păcate, BIOS-ul nu cunoaşte mecanismele legate de fişiere şi directoare, ci operează doar la nivel de blocuri (sectoare). Dacă se doreşte o operație pe disc, utilizând BIOS-ul, trebuie ştiut exact locul de pe disc (sectorul) unde se află fişierul. Pe de altă parte, utilizând DOS-ul se poate opera cu nume de fişiere, în locul adreselor de pe disc. DOS-ul păstrează informații asupra localizării fişierului pe disc şi apelează rutina ROM-BIOS pentru citirea/scrierea blocurilor respective. Interfața de nivel ridicat reduce efortul programatorului necesar pentru accesul la date.

Utilizarea funcțiilor DOS este, în general, mai uşoară decât cea a funcțiilor BIOS echivalente, primele fiind mult mai independente de maşină. Întreruperile de la 20H până la 3FH sunt rezervate pentru operații DOS, după cum urmează:

INT 20H = terminare program; operația termină execuția unui program .com, reface adresele pentru CTRL+Break şi pentru erorile critice, goleşte bufferele şi redă controlul sistemului DOS.

INT 21H = solicitare funcție DOS, al cărei cod este transmis în AH şi este descrisă în detaliu în continuare.

INT 22H = adresa de terminare. Se copiază adresa acestei întreruperi în PSP-ul programului (la offset 0AH) când programul încărcător încarcă un program pentru execuție. La terminarea programului, operația transferă controlul la adresa întreruperii.

INT 23H = adresa pentru CTRL+Break. Este destinată pentru transferul controlului la o rutină DOS (prin PSP, offset 0EH) când se tastează CTRL+Break sau CTRL+C. Rutina termină execuția unui program sau a unui fişier „batch“. Un program poate înlocui această adresă cu aceea a unei rutine proprii, pentru a realiza acțiuni specifice fără terminarea programului.

INT 24H = manipulare erori critice. Este utilizată de sistem pentru a transfera controlul (prin PSP, offset 12H) când este recunoscută o eroare critică (de obicei la o operație cu discul sau cu imprimanta). Aceste ultime trei întreruperi, în mod normal, nu ar trebui să fie apelate de programul utilizator.

INT 25H = citire absolută de pe disc a unuia sau mai multor sectoare (este înlocuită de INT 21H, funcția 440DH, codul minor 61H).

INT 26H = scriere absolută pe disc, din memorie, a unuia sau mai multor sectoare (este înlocuită de INT 21H, funcția 440DH, codul minor 41H).

INT 27H = programul se termină, dar rămâne rezident. Determină ca un program .COM, după terminare, să rămână rezident în memorie (înlocuită de INT 21H, funcția 31H).

INT 2FH = întrerupere multiplexare. Aceasta implică o comunicație între programe, cum ar fi comunicarea stării unei imprimante, prezența unui driver de disc, sau comenzi sistem cum ar fi ASSIGN sau APPEND.

INT 33H = manipulare mouse, deci furnizează serviciile pentru manipulare mouse.

Vom prezenta câteva funcții DOS, INT 21H, mai des utilizate:

Funcția 0 – terminare execuție program:

```
xor     ah, ah
int     21h
```

Funcția 1 – citire caracter de la tastatură, cu ecou:

```
mov     ah, 1
int     21h     ; (AL) = codul ASCII al caracterului citit sau 0 daca este o
                ; tasta functionala, cum ar <Home>, <F1> sau <PgUp>
```

Funcția 2 – afişare caracter:

```
mov     dl, cod_ASCII_caracter
mov     ah, 2
int     21h
```

Funcțiile 3 şi 4 – intrare sau ieşire la un port de comunicare:

Cele două funcții permit citirea (în AL), respectiv scrierea (din DL) a unui caracter la un port, dar este preferată operația BIOS INT 14H.

Funcția 5 – scrie caracter la LPT:

```
mov     dl, caracter
mov     ah, 5
int     21h
```

Funcția 6 – intrare/ieşire direct la consolă:

Această funcție poate transfera orice caracter sau cod de control fără interferență cu DOS-ul. Sunt două versiuni, una pentru intrare şi alta pentru ieşire. Pentru intrare se încarcă 0FFH în DL. Dacă nu există un caracter în bufferul tastaturii, operația pune ZF=1 şi nu aşteaptă intrarea. Dacă există un caracter, acesta este memorat în AL şi ZF=0. Operația este fără ecou şi nu verifică eventualele combinații <Ctrl>+<Break> sau <Ctrl>+PrtSc. Dacă AL este diferit de 0, este codul ASCII al unui caracter, dacă însă este 0, atunci este o tastă cu funcție extinsă (cum ar fi <Home>, <F1> sau <PgUp>) Pentru ieşire, caracterul se încarcă în DL (însă nu 0FFh)

```
mov     dl, caracter      ; sau 0FFh pentru iesire
mov     ah, 6
int     21h
; pentru iesire se testeaza ZF, dupa care se ia valoarea din AL, daca ZF=0.
```

Funcția 7 – citire de la tastatură, fără ecou şi fără interpretare break (<Ctrl> + <Break>):

```
mov     ah, 7
int     21h             ; (AL) = codul ASCII al caracterului citit
```

Funcția 8 – citire de la tastatură, fără ecou:

```
mov     ah, 8
int     21h             ; (AL) = codul ASCII al caracterului citit
```

Funcția 9 – afişarea unui şir de caractere, ce se termină cu „$“:

```
lea     dx, sir         ; (DS:DX)=adresa de inceput a sirului de caractere
mov     ah, 9
int     21h
```

Funcția 0Ah – citire şir de caractere:

Această funcție acceptă date de la tastatură, dar necesită o listă de parametri. Primul care trebuie cunoscut este numărul maxim de caractere (date) de intrare care nu mai poate fi introdus, valoare la apariția căreia se anunță sonor utilizatorul. Un al doilea parametru, actualizat de funcție, va conține numărul de caractere efectiv introduse. Lista de parametri constă din următoarele elemente:

1. Prima intrare furnizează numele listei de parametri, în forma LABEL BYTE.
2. Primul octet din listă conține limita maximă a numărului de caractere de intrare (între 0 şi 255).
3. Al doilea octet este rezervat pentru memorarea numărului efectiv de caractere tastate.
4. În continuare, este rezervată memorie pentru memorarea caracterelor.

În concluzie, o astfel de listă poate fi definită astfel:

```
param_list      label   byte
        dim_max         db      50      ; dimensiunea spatiului de memorie alocat
        dim_efectiv     db      ?       ; numarul efectiv de caractere citite
        dat             db      50 dup (?); aici se depun caracterele
```

Apelul funcției se realizează astfel:

```
lea     dx, param_list          ; (DS:DX) = adresa listei descrisa anterior
mov     ah, 0Ah
int     21h
```

Şirul de caractere tastat se va termina cu tasta <Enter>, al cărei cod nu este numărat, dar este transferat în zona de date. În cazul în care contorul numărului de caractere în momentul apăsării tastei <Enter> este mai mic decât limita numărului maxim de caractere, va fi depus în listă ca al doilea octet (parametru). Dacă se tastează mai multe caractere decât limita maximă stabilită, vor fi memorate doar primele caractere tastate care, alături de codul tastei <Enter>, 0Dh, se încadrează în zona de memorie rezervată (deci mai puțin cu unul decât numărul maxim de caractere specificat în lista de parametri). În acest capitol este prezentat un exemplu de utilizare a acestei funcții; programul citaftxt.asm citeşte un text de maximum 30 de caractere şi îl afişează pe centrul ecranului, program care se execută ciclic până la citirea unui text vid.

Funcția 0Bh – determinare stare tastatură:

```
mov     ah, 0Bh
int     21h     ; daca (AL) = 0FFH, atunci exista un caracter in buffer
```

```
                    ; daca (AL) = 00H, atunci nu exista nici un caracter
                    ; in buffer
```

Funcția 0Ch – şterge bufferul tastaturii şi invocă funcția din AL:

```
mov     ah, 0Ch         ; aceasta operatie poate fi utilizata in asociere cu
mov     al, nr_functie  ; functiile: 01h, 06h, 07h, 08h sau 0Ah
mov     dx, mem_KBD; zona de citire date
int     21h
```