ARHITECTURA SISTEMELOR DE CALCUL
LUCRAREA DE LABORATOR NR. 1

Controlul resurselor unui sistem de calcul utilizînd limbaje de nivel înalt şi funcţii de sistem de operare

Scopul lucrării

a) Studiul modalităţilor de control prin program a resurselor unui sistem de calcul : tastatură, display, disk. Exemple.
b) Studiul posibilităţilor de lucru cu porturile de intrare - ieşire şi utilizarea sistemului de întreruperi prin intermediul limbajelor de programare de nivel înalt. Exemple.
c) Studiul unor elemente de sistem de operare (funcţii DOS).Exemple.

1.1. Introducere

Resursele unui sistem de calcul constau în : unitatea centrală de prelucrare (CPU), sistemul de memorie (memorie principală, memorie de masă - disk, bandă magnetice) şi interfeţe de intrare - ieşire (pentru introducerea şi afişarea datelor). Toate aceste resurse concură la îndeplinirea sarcinii sistemului de calcul : efectuarea volumului de calcule asociate cu o anumită aplicaţie.

Resursele sistemului de calcul trebuie gestionate în aşa fel încît să se obţină performanţele maxime din punctul de vedere al vitezei de prelucrare, necesităţilor de memorare, etc.

Gestiunea resurselor sistemului de calcul revine unui program de supervizare, denumit sistem de operare.

Sistemul de operare constituie un mediu pentru execuţia programelor de aplicaţie.

Funcţiile sistemului de operare sînt :

- controlul execuţiei programelor (încărcare, lansare, terminare)
- asigurarea interfeţei cu operatorul (independenţa programelor relativ la hardware-ul de intrare-ieşire)
- interfată cu operatorul uman
- tratare erori
- gestionarea resurselor sistemului
    - alocarea hardware a CPU, memoriei, porturilor I/O
    - protecţia resurselor împotriva accesului neautorizat
    - evidenţa utilizării resurselor

Componentele principale ale sistemului de operare sînt : **nucleul (kernel)** - alocă resursele,asigura protecţia, controlează interfeţele de intrare-ieşire (I/O) la nivel fizic, tratează întreruperile, **executivul** - controlează interfeţele I/O la nivel logic, gestionează sistemul de fişiere, planifică activităţile din sistemul de calcul, coordonează programele de aplicaţie şi **supervizorul** - asigură interfaţa cu operatorul uman, asigură legătura cu celelalte nivele ale sistemului de operare, contabilizează utilizarea resurselor.

1.2. Funcţii DOS uzuale

Sistemul de opr\erare cel mai larg răspîndit este MS-DOS. Principalele componente ale acestui sistem de operare sînt : BDOS (sistemul de operare de bază) şi BIOS (componentă ce asigură interfaţa cu dispozitivele I/O).

Tabelul următor ilustrează cîteva dintre funcţiile DOS (asociate componentei BDOS):

Nr. funcţie	*Specificare*	*Reg. intrare*	*Reg. ieşire*
01H	citire tastatura	AH=01	AL <-- caracter citit
02H	afişare pe display	AH=02 DL <-- caracter de afişat	
05H	tiparire la imprimanta	AH=05 DL <-- caracter de tipărit	
09H	afişare şir pe display	AH=09 DS:DX <-- adresa şirului (terminat cu '$')	
2AH	afişare dată	AH=2A	AL <-- ziua saptamînii CX <-- anul DH <-- luna DL <-- ziua lunii
2CH	afişare timp	AH=2C	CH <-- ora CL <-- minutul DH <-- secunda DL <-- sutimi de secundă

Există de asemenea funcţii pentru crearea unui director (fişier), ştergerea unui director (fişier), setare atribute fişiere, controlul dispozitivelor I/O, alocarea memoriei, execuţie şi încărcare programe, terminare programe.

În general, funcţiile DOS nu sînt utilizate direct ; controlul resurselor sistemului de calcul se realizează mai comod prin intermediul unor proceduri (sau funcţii) asociate unor limbaje de programare de nivel înalt (Turbo Pascal, Turbo C, Turbo C++ ,etc.). Funcţiile DOS pot fi utilizate în situaţiile care necesită timpi de execuţie mici ; în acest caz apelul funcţiilor DOS se realizează în limbaj de asamblare, urmărind registrele implicate (ca în tabelul anterior).

1.3. Exemple de funcţii C pentru controlul resurselor sistemului de calcul

bdos : Apel functii DOS

Declaratie: int bdos(int dosfun, unsigned dosdx, unsigned dosal);

Param.	Semnificatie
dosfun	Defineste functia DOS
dosdx	Valoarea registrului DX
dosal	Valoarea registrului AL

Intoarce valoarea actualizata a registrului AX

---

bioskey : Lucrul cu tastatura utilizind in mod direct BIOS

Declaratie : int bioskey(int cmd);

cmd	Functie si valoare intoarsa
0	Intoarce tasta apasata Daca cei mai putin semnificativi 8 biti sint nenuli atunci valoarea este caracter ASCII;in caz contrar bitii mai semnificativi reprezinta codul extins al tastei
1	Testeaza daca s-a apasat o tasta; daca nu s-a apasat intoarce 0; daca s-a apasat CTRL-BRK intoarce -1; in caz contrar intoarce valoarea tastei.
2	Testeaza taste de control astfel :

Bit	*Value*	*Semnificatie*
0	0x01	Shift Dreapta tastat
1	0x02	Shift Stinga tastat
2	0x04	Ctrl tastat
3	0x08	Alt tastat
4	0x10	Scroll Lock activ
5	0x20	Num Lock activ
6	0x40	Caps activ
7	0x80	Insert activ

---

biosdisk : Utilizare disk cu ajutorul BIOS (in mod direct)

Declaratie : int biosdisk(int cmd, int drive, int head, int track, int sector,int nsects, void *buffer);

NOTA: biosdisk opereaza sub nivelul fisierelor. Se pot distruge continutul fisierelor si al directoarelor pe hard disk !!!

Param.	Semnificatie
cmd	Indica operatia de efectuat
drive	Specifica drive-ul ce va fi utilizat
head	Specifica sectorul de start
sector	
nsects	Numar de sectoare pentru transfer (1 sector = 512 octeti)
buffer	Adresa in memorie (pentru transfer)

Valoare intoarsa:

Operatie cu succes : octetul superior = 0;
octetul inferior contine numarul de sectoare (citite, scrise, verificate)
Operatie cu eroare : octetul superior = una din valorile

Val.	Descriere
0x01	Comanda eronata
0x02	Adresa inexistenta
0x03	Incercare de scriere pe disk protejat
0x04	Sector inexistent
0x05	Eroare de reset (hard disk)
0x06	Disk-ul a fost schimbat de la ultima operatie
0x07	Eroare de drive
0x0A	Sector defect
0x0B	Pista defecta
0x0C	Pista inexistenta
0x10	Eroare de verificare la citire CRC/ECC
0x11	Eraore CRC/ECC corectata
0x20	Controler Defect
0x40	Operatie de cauatre esuata
0xAA	Disk- ul nu e gata
0xE0	Eroare de stare

---

biosprint : Tiparire la imprimanta utilizind BIOS in mod direct

Declaratie : int biosprint(int cmd, int abyte, int port);

Arg.	Semnificatie
abyte	Caracterul de tiparit;valoare intre 0 si 255.

cmd	Functia imprimantei
0	Tipareste caracterul din abyte
1	Initializeaza imprimanta
2	Citeste starea imprimantei
	Daca cmd = 1 or 2 , abyte este ignorat.
port	Identifica imprimanta : 0 = LPT1, 1 = LPT2, etc.

Starea imprimantei este obtinuta astfel :

Bit	Valoare	Starea imprimantei
0	0x01	Time out
3	0x08	Eroare I/O
4	0x10	Selectata
5	0x20	Lipsa hirtie
6	0x40	Acknowledge
7	0x80	Not busy

---

inport / inportb - citeste un cuvint/octet de la un port
outport / outportb - scrie un cuvint/octet la un port

Declaratii :

```
unsigned      inport  (unsigned portid);
unsigned char inportb (unsigned portid);
void          outport (unsigned portid, unsigned      value);
void          outportb(unsigned portid, unsigned char value);
```

Citirea unui cuvint se efectueaza astfel :octetul inferior de la adr , iar cel superior de la adr+2.(Portul de intrare de 8 biti,conectat pe bus-ul de date inferior)

Scrierea unui cuvint se efectueaza astfel :octetul inferior al cuvintului la adr, iar cel superior la adr+1.(Portul de iesire de 16 biti).

Argument	Semnificatie
portid	Adresa portului
value	Cuvintul / octetul citit/scris

1.4. Exemple de programe

{ASCL1.C - lucrul cu fisiere si cu grafica }

```
#define n           100
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <dos.h>
#include <graphics.h>

int gdriver = DETECT, gmode, errorcode;

void main(void)
{
FILE *fis;
int i;
double sgn[n];
double a=0.001;
char str[9];
int s,s1,s2;
double b,j;

clrscr();
fis=fopen("sgn.dat","wt");
/*deschide fisierul "sgn.dat", pentru scriere in mod text */
for(i=1;i<n+1;i++)
        {
//calcule numerice
        j=0.01*i;
        b=1/(1-a*j*j);
        sgn[i]=(1/j)*log(b);
        sprintf(str,"%f\n",sgn[i]);
// converteste sgn[i] din float in sir de caractere
        fwrite(&str,sizeof(str),1,fis);
// scrie in fisier
        }
fclose(fis); // inchide fisierul
initgraph(&gdriver, &gmode, "c:\\bc\\bgi\\");
  s=10; //factor de scala
//afisare grafica  (mod grafic 640 x 480 pixeli)
 for(i=1;i<n-1;i++)
        {
        s1=480-100000*sgn[i];
        s2=480-100000*sgn[i+1];
        line(i*s,s1,(i+1)*s ,s2);
        }
  getch();
  closegraph();
}
```

{ASCL2.C - lucrul cu întrruperi}

```
#include <stdio.h>
#include <dos.h>
#include <conio.h>
```

```
#define INTR 0X1C    /* Intrerupere de la timer PC - la 50ms */
void interrupt ( *oldSCI)(void);
//rutina de servire a intreruperii anterioara

int count1=0,count=0,flag=0;

//noua rutina de servire a intreruperii
void interrupt newSCI(void)
{
  //numara 20 intrruperi - 1 sec
  count++;
  if (count==20) {count1++;flag=1;count=0;}

/* apel old SCI */
  oldSCI();
}

void main(void)
{
        clrscr();
        printf("Start asteptare intreruperi\n");

/* salveaza vectorul de intrerupere vechi*/
  oldSCI = getvect(INTR);

/* scrie noul vector se intrerupere*/
  setvect(INTR, newSCI);

/* bucla de asteptare intreruperi pina cind counter > 20 */
  while (count1 < 20)
  if (flag==1) { printf("%d\n",count1);flag=0;}

        printf("Asteptare intreruperi incheiata\n");
        printf("Tastati orice tasta pentru a termina programul\n");

/* reface vechiul vector de intrerupere*/
  setvect(INTR, oldSCI);
  while (!kbhit());
}
```

{ASCL3.C - funcţii DOS}

```
#include <stdio.h>
#include <dos.h>

/* citeste unitatea de disc curenta ca 'A', 'B', ... */
char current_drive(void)
{
  char disc;

  /* citeste discul curent ca 0, 1, ... */
  disc = bdos(0x19, 0, 0);
  return('A' + disc);
}
```

```
void main(void)
{
  printf("Unitatea de disc curenta este %c \n", current_drive());
}
```

{ASCL4.C - lucrul cu tastatura (utilizare directa BIOS)}

```
#include <stdio.h>
#include <bios.h>
#include <ctype.h>

#define RIGHT  0x01
#define LEFT   0x02
#define CTRL   0x04
#define ALT    0x08

void main(void)
{
  int key, modifiers;

  /* functia 1 intoarce 0 pina cind se apasa o tasta */
  while (bioskey(1) == 0);

  /* functia 0 intoarce codul tastei apasate */
  key = bioskey(0);

  /*functia 2 determina daca au fost utilizate taste de control
  SHIFT stinga sau SHIFT dreapta, CTRL, ALT */
  modifiers = bioskey(2);
  if (modifiers)
  {
    printf("[");
    if (modifiers & RIGHT) printf("RIGHT");
    if (modifiers & LEFT)  printf("LEFT");
    if (modifiers & CTRL)  printf("CTRL");
    if (modifiers & ALT)   printf("ALT");
    printf("]");
  }
  /* afisaeza caracterul citit de la tastatura */
  //test daca tasat apasata este caracter alfanumeric
  if (isalnum(key & 0xFF))
    printf("'%c'\n", key);
  else
    printf("%#x\n", key);
    //afisare in hexazecimal (forma alternata, cu prefixul 0x)
}
```

{ASCL5.C - funcţii BIOS - disk}

```
#include <bios.h>
#include <stdio.h>

void main(void)
{
  int result;
  char buffer[512];
```

```
  printf("Test daca drive-ul a: e pregatit (Ready) \n");
  result = biosdisk(4,0,0,0,0,1,buffer);
  result &= 0x02;
  (result) ? (printf("Drive A: Ready\n")) :
              (printf("Drive A: Not Ready\n"));

}
```

{ASCL6.C - funcţii BIOS - imprimanta}

```
#include <stdio.h>
#include <conio.h>
#include <bios.h>

void main(void)
{
  #define STATUS  2    /* comanda de citire a starii imprimantei*/
  #define PORTNUM 0    /* identificator port pentru LPT1 */

  int status, abyte=0;

  printf("Opriti imprimanta. Apasati orice tasta pentru a continua\n");
  getch();
  status = biosprint(STATUS, abyte, PORTNUM);
  if (status & 0x01)
    printf("Stare : time out \n");
  if (status & 0x08)
    printf("Stare : Eroare I/O \n");
  if (status & 0x10)
    printf("Stare : Imprimanta selectata \n");
  if (status & 0x20)
    printf("Stare : Lipsa hirtie \n");
  if (status & 0x40)
    printf("Stare : Acknowledge \n");
  if (status & 0x80)
    printf("Stare : Not busy \n");

}
```

{ASCL7.C - lucrul cu porturile}

```
#include <stdio.h>
#include <dos.h>

void main(void)
{
  int port = 0;
  int value = 'C';

  outport(port, value);
  printf("Valoarea %d a fost scrisa la portul de adresa %d\n", value, port);
}
```

1.5. Desfăşurarea lucrării

a) Să se studieze exemplele de la punctul 1.4. Să se verifice corectitudinea funcționării programelor.

b) Să se realizeze un program ce afisează pe ecran codul unei taste apăsate, folosind funcții DOS.

c) Să se realizeze un program de tipărire la imprimantă folosind funcții DOS.

d) Să se realizeze un program de tipărire la imprimantă folosind funcții C.

e) Să se scrie un program ce implementează un ceas digital care afişează pe ecran minutele şi secundele, utilizînd sistemul de întreruperi (ceasul de timp real al PC).

f) Să se scrie un program similar celui de la punctul e) , dar utilizînd funcții DOS. (afişare an,luna, zi, ziua săptămînii, ore , minute, secunde).

h) Să se studieze programul demostartiv BGIDEMO.C din kit-ul de instalare Borland C ++.