
Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Operativni sistemi 1
Nastavnik: prof. dr Dragan Milićev
Odsek: Softversko inženjerstvo, Računarska tehnika i informatika
Kolokvijum: Treći, avgust 2024.
Datum: 1. 9. 2024.

Treći kolokvijum iz Operativnih sistema 1

Kandidat: _____

Broj indeksa: _____ *E-mail:* _____

Kolokvijum traje 90 minuta. Dozvoljeno je korišćenje literature.

Zadatak 1 _____/10 *Zadatak 3* _____/10
Zadatak 2 _____/10

Ukupno: _____/30 = _____% = _____/10

Napomena: Ukoliko u zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku, da je uokviri (da bi se lakše prepoznala prilikom ocenjivanja) i da nastavi da izgrađuje preostali deo svog odgovora na temeljima uvedene pretpostavke. Ocenjivanje unutar potpitanja je po sistemu "sve ili ništa", odnosno nema parcijalnih poena. Kod pitanja koja imaju ponuđene odgovore treba **samo zaokružiti** jedan odgovor. Na ostala pitanja odgovarati **čitko, kratko i precizno**.

1. (10 poena) Ulaz/izlaz

Zahtev za operacijom sa nekog ulaznog uređaja predstavljen je strukturom `IORequest`. Date su deklaracije pokazivača preko kojih se može pristupiti registrima kontrolera ovog uređaja, kao i simboličkih konstanti za vrednosti u njima, sa značenjem objašnjenim na predavanjima. Uređaj generiše prekid za svaki spreman podatak, s tim što se taj prekid može maskirati i demaskirati funkcijama čiji su potpisi dati dole.

Uređaj svoje podatke isporučuje u naletima koji pristižu u nepredvidivim intervalima. Kada dođe jedan nalet neodređene dužine (tj. promenljivog broja pojedinačnih podataka), uređaj vrlo brzo isporučuje jedan po jedan podatak, da bi onda napravio pauzu nepredvidivog trajanja, nakon koje opet stiže nov nalet podataka, itd. sve dok se ne završi prenos celog niza podataka iz jednog zahteva. Zato ulaznu operaciju treba obavljati na sledeći način. Uređaj treba prozivati (*polling*), u očekivanju da naredni podatak stigne brzo nakon prethodnog. Međutim, kako se može dogoditi da je jedan nalet završen i da nastupa pauza, ovo prozivanje treba raditi najviše `MAX_POLL_CNT` puta. Ako se podatak ne pojavi nakon `MAX_POLL_CNT` ispitivanja bita spremnosti, tok kontrole koji izvršava prozivanje treba suspendovati („uspavati“), dozvoliti prekid uređaju kojim će on onda signalizirati spremnost narednog podatka kojim počinje nov nalet, a koji onda ponovo treba učitavati prozivanjem itd. Za uspavljivanje koristiti klasu `Event` koja realizuje događaj sa uobičajenim operacijama `wait` i `signal` koje se mogu pozivati iz prekidnih rutina i odgovarajućim konstruktorom.

```
typedef volatile unsigned int REG;
REG* ioCtrl = ...; // control register
REG* ioStatus = ...; // status register
REG* ioData = ...; // data register

const REG C_START = 0x...;
const REG C_STOP = 0x...;
const REG C_READY = 0x...;

void mask_intr();
void unmask_intr();

struct IORequest {
    REG* buffer; // Data buffer
    size_t len; // Size of the buffer
    void (*signal)(); // Call back upon I/O completion
    IORequest* next; // Next in the list
};
```

Potrebno je implementirati operaciju `transfer`, zajedno sa odgovarajućom prekidnom rutinom `device_ready` za prekid od uređaja, za obradu jednog zahteva.

```
void transfer (IORequest* req);
interrupt void device_ready ();
```

Rešenje:

2. (10 poena) Fajl sistem

Tekući direktorijum Unix interpretera komandne linije je inicijalno prazan. U komandnoj liniji se redom zadaju navedene komande. U datim proredima između nekih komandi napisati šta će biti izlaz nakon izvršene komande navedene neposredno ispred tog proreda. Objašnjenje komandi:

- `mkdir directory`: pravi navedeni direktorijum sa podrazumevanim pravima pristupa; pretpostaviti da su ta podrazumevana prava pristupa dovoljna da tekući korisnik uradi sve komande;
- `cat`: ukoliko nije naveden ulazni fajl, ovaj sistemski program sadržaj koji daje na svoj standardni izlaz uzima sa svog standardnog ulaza; pretpostaviti da je nakon prve ove komande korisnik uneo znakove `xyz` sa tastature, kao i kontrolni znak za kraj ulaza;
- `ln original_filename link_name`: pravi tvrdi vezu `link_name` ka fajlu `original_filename`;
- `rm node`: uklanja čvor.

```
mkdir a
ls
```

```
cat >b
ls
```

```
cat b
```

```
mkdir c
ls
```

```
ln b a/d
ls
```

```
cat b>a/e
ls a
```

```
ln a/e c/f
cat c/f
```

```
rm b
cat a/d
```

```
ls
```

```
rm c/f
ls a
```

```
cat a/e
```

3. (10 poena) Fajl sistem

Neki fajl sistem primenjuje kombinovanu tehniku indeksirane alokacije sadržaja fajla. U FCB fajla polje `singleIndex` predstavlja direktni indeks kao niz od `SingleIndexSize` elemenata, pri čemu svaki element sadrži broj fizičkog bloka sa sadržajem fajla (ili 0, ako je neiskorišćen). Ako veličina sadržaja fajla preraste veličinu podržanu ovim indeksom, naredni blokovi sadržaja fajla (preko ove veličine) indeksiraju se indeksom u dva nivoa. Za te potrebe, polje `dblIndex` u FCB sadrži broj bloka na disku u kom je indeks drugog nivoa. Taj blok sa indeksom drugog nivoa sadrži `DblIndexSize` ulaza sa brojevima blokova sa sadržajem fajla, odnosno 0 ako ulaz nije iskorišćen. Polje `size` u FCB sadrži veličinu fajla u bajtovima. Veličina bloka je `BlockSize`. Svi ulazi u indeksu prvog nivoa i indeksu drugog nivoa (ako postoji) su sigurno postavljeni na validne vrednosti (broj bloka sa sadržajem ili 0).

Realizovati funkciju `truncateFile` koja treba da obriše (odseče) sadržaj iza zadate pozicije fajla čiji je FCB `dat`, tako da veličina sadržaja tog fajla postane `newSize` bajtova (ovaj argument sigurno ima ispravnu vrednost koja je manja ili jednaka prethodnoj veličini fajla):

```
void truncateFile (FCB* fcb, size_t newSize);
```

Na raspolaganju je funkcija koja vraća pokazivač na sadržaj bloka na disku sa datim brojem, učitano u keš:

```
void* getBlock (PBlock blkNo);
```

kao i funkcija koja blok sa datim brojem označava slobodnim:

```
void freeBlock (PBlock blkNo);
```

Rešenje: