
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 2026.
Datum: 6. 9. 2026.

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 = _____% = _____/15

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

Date su deklaracije pokazivača preko kojih se može pristupiti registrima DMA kontrolera:

```
typedef volatile unsigned int REG;
REG* dmaCtrl =...;    // DMA control register
REG* dmaStatus =...;  // DMA status register
REG* dmaAddr =...;    // DMA register for setting buffer address
REG* dmaBlk =...;     // DMA register for setting device block
```

Ovaj DMA kontroler upravlja nekim ulaznim blokovskim uređajem sa direktnim pristupom. Kontroleru može da se istovremeno zahteva do dva prenosa blokova zadavanjem adrese ulaznog bafera u memoriji i broja bloka na uređaju (blokovi su iste veličine) koji treba učitati. Kada je DMA u pasivnom stanju (završio je ranije prenose), nov prenos do dva bloka mu se zadaje na sledeći način: prvi upis u registre `dmaAddr` i `dmaBlk` zadaje prenos jednog bloka, a naredni upis u iste registre zadaje prenos drugog bloka. Nakon toga treba pokrenuti prenose upisom 1 u kontrolni registar; ako se to uradi nakon prvog upisa u navedena dva registra, DMA će obavljati prenos samo jednog bloka. Kada završi oba zadata prenosa, bilo ispravno ili u slučaju greške pri bilo kom od njih, DMA kontroler zaustavlja prenos, generiše prekid procesoru i prebacuje se ponovo u pasivno stanje. U statusnom registru najniži bit je bit završetka transfera (*TransferComplete*), a bit do njega bit greške (*Error*) u nekom od prenosa. Završetak oba prenosa postavlja bit *TransferComplete* i generiše zahtev za prekid procesoru.

Zahtevi za ulaznim operacijama na ovom uređaju vezani su u jednostruko ulančanu listu. Zahtev ima sledeću strukturu:

```
struct IORequest {
    REG* buffer; // Data buffer (block)
    REG blk; // Device block number
    int status; // Status of operation
    IORequest* next; // Next in the list
    Task* task; // Task waiting for the transfer
};
```

Na prvi zahtev u listi pokazuje globalni pokazivač `ioHead`, a na poslednji `ioTail`. Operacijom `transfer` kernel stavlja jedan zahtev u ovu listu i po potrebi pokreće transfer ako je DMA kontroler u pasivnom stanju. Kada se završe zadati prenosi, potrebno je u polje `status` date strukture preneti status svake od najviše dve završene operacije (0 – ispravno završeno do kraja, -1 – greška), probuditi zadatak (`task`) koji je taj zahtev postavio pozivom njegove operacije `resume` i pokrenuti nove prenose za sledeće zapise u listi ako ih ima, a zahteve izbaciti iz liste. Tokom izvršavanja koda kernela nema preotimajna i svi prekidi su maskirani, tako da ne treba brinuti o konfliktima zbog konkurentnosti.

Potrebno je napisati funkciju `transfer`, zajedno sa odgovarajućom prekidnom rutinom `dmaInterrupt` za prekid od DMA kontrolera.

```
void transfer (IORequest* request);
interrupt void dmaInterrupt ();
```

Rešenje:

2. (10 poena) Fajl sistem

Dat je kratak izvod iz dokumentacije Linux sistemskih poziva za pravljenje tvrdih i simboličkih veza:

```
#include <unistd.h>
int link(const char *oldpath, const char *newpath);
int symlink(const char *target, const char *linkpath);
```

link() creates a new link (also known as a hard link) to an existing file. If newpath exists, it will not be overwritten.

This new name may be used exactly as the old one for any operation; both names refer to the same file (and so have the same permissions and ownership) and it is impossible to tell which name was the "original".

symlink() creates a symbolic link named linkpath which contains the string target. Symbolic links are interpreted at run time as if the contents of the link had been substituted into the path being followed to find a file or directory.

Symbolic links may contain .. path components, which (if used at the start of the link) refer to the parent directories of that in which the link resides.

A symbolic link (also known as a soft link) may point to an existing file or to a nonexistent one; the latter case is known as a dangling link.

Napisati deo C koda kojim se, korišćenjem ovih sistemskih poziva, postiže isto što i sledećim komandama iz komandne linije (podsetnik `ln target linkpath`), za isti tekući direktorijum; ignorisati greške:

```
ln /home/docs/bar foo
cd /home/docs
ln -s bar /home/bar
```

Rešenje:

3. (10 poena) Fajl sistem

Neki fajl sistem koristi listu prava pristupa (engl. *access control list*, ACL) za definisanje i proveru prava pristupa fajlu na sledeći način. U klasi `FCB` koja predstavlja FCB učitani u memoriju postoji polje `acl` koje je niz veličine `ACL_SIZE` elemenata tipa strukture `ACLEntry`. Svaki element niza može da sadrži definiciju prava pristupa za jednog korisnika definisanog vrednošću polja `uid` tipa `uid_t`; ako je ova vrednost 0, ulaz je nekorišćen („prazan“). Polje `rxw` u ovoj strukturi u odgovarajućim bitima definiše prava pristupa datom korisniku.

Implementirati funkciju `FCB::checkRight` koja treba da proveri da li korisnik dat parametrom `uid` ima sva prava definisana parametrom `rxw` u kojima su postavljeni biti za zahtevana prava na istim mestima kao i u polju `rxw` strukture `ACLEntry` (vraća `true` ako i samo ako su sva zahtevana prava odobrena).

```
struct ACLEntry {
    uid_t uid; short rxw;
};

class FCB {
public:
    ...
    bool checkRight (uid_t uid, short rxw) const;
    ...
private:
    ...
    ACLEntry acl[ACL_SIZE];
};
```

Rešenje: