Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Operativni sistemi 1 (SI2OS1, IR2OS1)
Nastavnik: prof. dr Dragan Milićev
Odsek: Softversko inženjerstvo, Računarska tehnika i informatika
Kolokvijum: Prvi, septembar 2013.
Datum: 30.8.2013.

Prvi kolokvijum iz Operativnih sistema 1

Kandidat: _____

Broj indeksa: _____ *E-mail:*_____

Kolokvijum traje 1,5 sat. Dozvoljeno je korišćenje literature.

Zadatak 1	_____/10	*Zadatak 3*	_____/10
Zadatak 2	_____/10	*Zadatak 4*	_____/10

Ukupno: _____/40 = _____% = _____/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)

Date su deklaracije pokazivača preko kojih se može pristupiti registrima jednog ulaznog uređaja:

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

U upravljačkom registru najniži bit je bit *Start* kojim se pokreće uređaj, a u statusnom registru najniži bit je bit spremnosti (*Ready*), a bit do njega bit greške (*Error*). Svi registri su veličine jedne mašinske reči (tip `unsigned int`).

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

```
struct IORequest {
  REG* buffer; // Data buffer (block)
  unsigned int size; // Buffer (blok) size
  void (*callBack)(IORequest*); // Call-back function
  int status; // Status of operation
  IORequest* next; // Next in the list
};
```

Kada se završi prenos zadat jednim zahtevom, potrebno je pozvati funkciju na koju ukazuje pokazivač `callBack` u tom zahtevu, sa argumentom koji ukazuje na taj zahtev. Ovu funkciju implementira onaj ko je zahtev postavio i služi da mu signalizira da je zahtev obrađen. U polju `status` date strukture treba preneti status završene operacije (0 – ispravno završeno do kraja, -1 – greška). Obrađeni zahtev ne treba brisati iz liste (to je odgovornost onog ko je zahtev postavio).

Potrebno je napisati kod operacije `transfer()`, zajedno sa odgovarajućom prekidnom rutinom za prekid od uređaja `ioInterrupt()`, koja obavlja sve prenose zadate zahtevima u listi na čiji prvi zapis ukazuje argument `ioHead` tehnikom programiranog ulaza korišćenjem prekida.

```
void transfer (IORequest* ioHead);
interrupt void ioInterrupt ();
```

Rešenje:

2. (10 poena)

Neki računar podržava segmentno-straničnu organizaciju virtuelne memorije, pri čemu je virtuelni adresni prostor veličine 8GB, adresibilna jedinica je 16-bitna reč, a fizički adresni prostor je veličine 512MB. Maksimalna veličina segmenta je 32MB, a stranica je veličine 512B.

a)(5) Prikazati logičku strukturu virtuelne i fizičke adrese i označiti širinu svakog polja.

b)(5) Napisati heksadecimalni kod fizičke adrese u koju se preslikava virtuelna adresa u segmentu 45h, stranici broj 23h u tom segementu, reč broj FFh u toj stranici, ako se ta stranica preslikava u okvir broj FF00h.

3. (10 poena)

a)(7) Na asembleru nekog RISC procesora sa *load/store* arhitekturom, poput onog opisanog na predavanjima, napisati prevod sledeće rekurzivne C funkcije koja izračunava najveći zajednički delilac (engl. *greatest common divisor*, GCD) dva data nenegativna cela broja x i y Euklidovim algoritmom, pri čemu je uvek $x \geq y \geq 0$ (pretpostaviti da su argumenti uvek ovakvi i ispravni). Funkcija `remainder(x,y)` je implementirana i vraća ostatak pri deljenju x sa y.

```
unsigned int gcd (unsigned int x, unsigned int y) {
  if (y==0) return x;
  else return gcd(y,remainder(x,y));
}
```

b)(3) Da li je ova funkcija bezbedna za uporedne pozive iz konkurentnih niti? Obrazložiti.

Rešenje:

4. (10 poena)

U nekom operativnom sistemu postoji sistemski poziv

```
int thread_create(void (*)(void*), void*)
```

koji kreira nit nad funkcijom na koju ukazuje prvi argument. Ta funkcija prima jedan argument tipa `void*` i ne vraća rezultat. Novokreirana nit poziva tu funkciju sa stvarnim argumentom jednakim drugom argumentu ovog sistemskog poziva.

Stablo u kome svaki čvor sadrži jedan podatak tipa `Data` i proizvoljno mnogo potomaka realizovano je pomoću sledeće strukture čvora:

```
struct Node {
  Data data;
  Node* children_head;
  Node* next_sibling;
};

extern void process(Data*);
```

Potomci datog čvora uvezani su u jednostruko ulančanu listu na čiji prvi element ukazuje polje `children_head`. Na sledećeg u listi ukazuje polje `next_sibling`. Procedura `process` vrši obradu datog podatka u jednom čvoru.

Na jeziku C napisati proceduru `visit_node` koja, kada se pozove za koren stabla (ili bilo kog podstabla), obilazi to stablo rekurzivno na sledeći način: najpre obradi podatak u tom čvoru pozivom procedure `process`, a zatim kreira po jednu nit koja obilazi podstablo svakog podčvora počev od drugog u listi (ako postoji), a u istoj niti obilazi podstablo prvog podčvora (ako postoji).

Rešenje: