Prvi kolokvij iz predmeta *Operacijski sustavi*

3. svibnja 2017.

PISATI ČITKO $\Longrightarrow$ NEČITKO $\equiv$ KRIVO

1. (2) Koji su osnovni zadaci operacijskog sustava?

2. (2) Što je *kontekst dretve*?

3. (2) Navesti koje sve radnje mogu generirati prekide unutar procesora (prekidi čiji je uzrok izvođenje instrukcija).

4. (2) Navesti/opisati Lamportov algoritam međusobnog isključivanja.

5. (2) Skicirati graf mogućih stanja dretvi u jednostavnom modelu jezgre prikazanom na predavanjima.

6. U nekom sustavu sa sklopom za prihvat prekida javljaju se prekidi P_1 u 5., P_2 u 1. i 8. te P_3 u 3. ms. Prioritet prekida određen je brojem (P_3 ima najveći prioritet). Procedura za prihvat prekida (PP) traje 0,5 ms a procedura za povratak iz prekida (PiP) 0,5 ms. Obrada svakog prekida traje po 3 ms.
   a) (3) Grafički prikazati aktivnosti procesora u glavnom programu (GP), procedurama za obradu prekida (P_i) te procedurama za prihvat prekida (PP) i povratak iz prekida (PiP).
   b) (1) Opisati stanje sustava u trenutku t=10 ms (što procesor radi, stanje sklopa za prihvat prekida, stanje na stogu).

7. (4) Neka ulazno-izlazna naprava generira novi podatak svakih 10 ms. Usporediti opterećenje (u postocima procesorskog vremena) koje ta naprava stvara na sustav ukoliko se za prijenos podatka u spremnik koriste prekidi (uz prosječno 100 sabirničkih ciklusa za obradu takvog prekida) te sklop s izravnim pristupom spremniku (zanemariti trajanje obrade prekida koje taj sklop izaziva obzirom da su vrlo rijetki). Trajanje sabirničkog ciklusa jest 10 ns (1 ns = 10^{-9} sekundi).

   Rješenje:

   Prekidi:
   • jedan prekid svakih 10 ms
   • prekid traje: 100 ciklusa $\cdot$ 10 ns = 100 $\cdot$ 10 ns = 1 μs
   • prekidi troše: 1 μs / 10 ms = $10^{-6}/(10{\cdot}10^{-3}) = 10^{-4}$ = 0,0001 = 0,01 % ostaje 99,99 %

   DMA:
   • jedan prijenos svakih 10 ms
   • prijenos traje: 1 ciklus $\cdot$ 10 ns = 10 ns
   • prijenos troši: 10ns/10ms = $(10{\cdot}10^{-9})/(10{\cdot}10^{-3}) = 10^{-6}$ = 0,000001 = 0,0001 % ostaje 99,9999 %

8. (4) Neki posao podijeljen je u sedam zadataka Z_1 do Z_8. Poznato je da će rezultat biti ispravan ako se zadaci izvode slijedno: $Z_1 \rightarrow Z_2 \rightarrow Z_3 \rightarrow Z_4 \rightarrow Z_5 \rightarrow Z_6 \rightarrow Z_7 \rightarrow Z_8$. Svaki zadatak Z_i koristi domenu $D_i = M_{1+(i-1) \bmod 6}$ te kodomenu $K_i = M_{2+(i \cdot 2) \bmod 5}$. Npr. zadatak Z_7 koristi domenu $D_7 = M_1$ i kodomenu $K_7 = M_{2+(7 \cdot 2) \bmod 5} = M_6$. Nacrtati tablicu sa zadacima, njihovim domenama i kodomenama te odrediti maksimalno paralelni sustav zadataka uzimajući u obzir njihove domene i kodomene te međusobni odnos u lancu.

9. (4) Stanje nekog sustav u promatranom trenutku je: dretva D_3 je aktivna, D_2 je u redu pripravnih, D_5 u redu semafor `OSEM[1]`, te D_4 i D_1 u redu odgođenih: D_4 treba čekati još jedan kvant vremena, a D_1 ukupno još 3. Red pripravnih dretvi je uređen prema prioritetu. Prioritet dretve određen je njenim indeksom – dretva D_5 ima najveći prioritet. U tom stanju dogodi se prekid sata te nakon toga tada aktivna dretva pozove `Postavi_OSEM(1)`. Prikazati stanje sustava nakon svakog od ta dva poziva jezgrine funkcije.

   Rješenje:

redovi	početno stanje	nakon prekida sata	nakon PostaviOSEM(1)
Aktivna_D	D3	D4	D5
Pripravne_D	D2	D3 D2	D4 D3 D2
red OSEM[1]	D5	D5	-
Odgođene_D	D4(1) D1(2)	D1(2)	D1(2)

10. (4) U nekon sustavu posluživanja postoje dva službenika, ali jedan (zajednički) red u koji dolaze klijenti te čekaju na posluživanje. Kad dođe na red, klijent preda zahtjev slobodnom službeniku te ode. Službenik tada započinje s obradom tog zahtjeva. Nakon što službenik obradi zahtjev, signalizira da može primiti idućeg klijenta. Simulirati klijente (od pojave, ulaska u red, predaje zahtjeva te izlaska iz sustava) i službenike dretvema te ih sinkronizirati semaforima (binarnim i/ili općim) te dodatnim varijablama. U kodu dretve `klijent` treba biti vidljiva operacija `predaj_zahtjev(i)`, gdje je `i` oznaka službenika kojemu je zahtjev predan (1 ili 2), a u kodu dretvi `službenik_1` i `službenik_2` operacija `uzmi_i_obradi_zahtjev()`. Osigurati da se klijenti poslužuju prema redu prispijeća – ne smije se dozvoliti preskakanje reda (pretpostaviti da su redovi semafora složeni prema redu prispijeća). Navesti početne vrijednosti korištenih semafora i varijabli.

Jedno rješenje:

```
BSEM[KO].v = 1   // za kritični odsječak
BSEM[RED].v = 0  // za čekanje na slobodnog službenika
BSEM[S[1]].v = 0 // za signaliziranje prvom službeniku
BSEM[S[2]].v = 0 // za signaliziranje drugom službeniku
SL[2] = {1,1}    // stanje službenika 0-slobodan, 1-zauzet

službenik(i) { // ili službenik_1 i 2, uz zamjenu u tekstu
    ponavljaj {
        ČekajBSEM(KO)          // može i bez KO na ovom mjestu, ali ...
        SL[i] = 0              // označi da je slobodan
        PostaviBSEM(KO)

        PostaviBSEM(RED)       // "pozovi" klijenta
        ČekajBSEM(S[i])        // čekaj da klijet preda zahtjev
        uzmi_i_obradi_zahjev()
    }
}
klijent {
    ČekajOSEM(RED)             // čeka slobodnog
    ČekajBSEM(KO)              // ulaz u KO, bar jedan službenik je slobodan
    ako je SL[1] == 0 tada
        i = 1
    inače
        i = 2
    SL[i] = 1                  // odmah označi da je službenik zauzet
    predaj_zahtjev(i)
    PostaviBSEM(S[i])          // signaliziraj službeniku
    PostaviBSEM(KO)            // izlaz iz KO
}
```