OPERACIJSKI SUTAVI
(Pitanja za ispit)

1.

1 . Što je operacijski sustav?

Skup programa koji omogućavaju izvođenje osnovnih operacija na računalu – potpora raznovrsnim primjenskim programima

2. Koji su osnovni zadaci operacijskog sustava?

- operacijski sustav "nadoknađuje" sva ograničenja i nedostatke sklopovija i stvara privid stroja koji je mnogo prikladniji za korištenje
- olakšavanje uporabe računala.
- organizacija djelotvornog iskorištavanja svih dijelova računala
- Operacijski sustav se mora pobrinuti da se procesor prebacuje s izvođenja jednog niza instrukcija na drugi i podržati višeprogramski rad.
- Operacijski sustav mora svakom pojedinom programu omogućiti pristup do potrebnih mu datoteka i svih ostalih sredstava. Sva nužna sredstva operacijski sustav mora dodjeljivati pojedinim programima i oduzimati ih od drugih tako da ona budu što je moguće bolje iskorištena.
- Operacijski sustav mora omogućiti ostvarenje komunikacije između računala ako su ona spojena u mrežu.

3. Navesti osnovne dijelove operacijskog sustava.

Osnovni dijelovi operacijskog sustava

1. upravljanje vanjskim jedinicama,
2. upravljanje spremničkim prostorom ("memorijom"),
3. upravljanje dretvama/procesima,
4. mehanizmi komunikacije i sinkronizacije dretvi/procesa,
5. datotečni sustav,
6. upravljanje sigurnošću,
7. sučelje prema aplikacijama (API),
8. sučelje prema korisniku (neizravno preko vanjskih jedinica).

4. Što je to sučelje?

Način postavljanja zahtjeva operacijskom sustavu, kao i izgled povratnih poruka operacijskog sustava, mora biti dogovoren. Utvrđeni način takva komuniciranja zovemo **sučeljem**.

Naziv sučelja koristi se, općenito, za čvrsto dogovoreni način uspostavljanja veze izmedu nekih, inače razdvojenih, cjelina.

2.

1. Čime su određena svojstva i ponašanje procesora?

Osnovna svojstva i ponašanje procesora određeni su skupom registara i skupom instrukcija koje procesor može obaviti. Registri služe za pohranjivanje svih informacijskih sadržaja koji ulaze i izlaze iz procesora i u njemu se transformiraju. Skup instrukcija određen je izvedbom aritmetičko-logičke i upravljačke jedinke procesora.

2. Navesti osnovni skup registara procesora.

- Adresni međuregistar
- Podatkovni međuregistar
- Instrukcijski registar
- Programsko brojilo (PC)
- Registar kazaljke stoga (SP)
- Registar stanja (SR)
- Registri opće namjene (R)

3. Što je to sabirnički ciklus?

Period u kojem se obavlja jedno čitanje ili pisanje u radni spremnik.
Sabirnički ciklus je onaj podciklus instrukcije koji uključuje promet, odnosno općenito aktivnost, na vanjskim sabirnicama procesora.

U svakom instrukcijskom ciklusu mora postojati bar jedan sabirnički ciklus –ciklus čitanja iz memorije (ciklus pribavljanja instrukcije iz memorije).

4. U pseudokodu napisati što procesor trajno radi.

ponavljati {

dohvatiti iz spremnika instrukciju na koju pokazuje programsko brojilo; dekodirati instrukciju, odrediti operaciju koju treba izvesti; povečaj sadržaj programskog brojila tako da pokazuje na sljededu instrukciju; odrediti odakle dolaze operandi i kuda se pohranjuje rezultat; operande dovesti na aritmetičko-logičku jedinicu, izvesti zadanu operaciju; pohraniti rezultat u odredište;

} dok je (procesor uključen);

5. Što je kontekst dretve

Sadržaj trenutne dretve pohranjen u registrima procesora; svi registri osim programskog brojila.

6. Što se zbiva pri izvođenju instrukcije za poziv potprograma?

Ponavljati {
iz spremnika dohvatiti instrukciju na koju pokazuje programsko brojilo; Dekodirati instrukciju, odrediti operaciju koja se treba izvest; Povećati programsko brojilo, da pokazuje na sljedeću instrukciju; Ako je (dekodirana instrukcija poziv potprograma) {
Pohraniti sadržaj programskog brojila na stog;

Smanjiti sadržaj registra SP, tako da pokazuje na sljedede prazno mjesto; Iz adresnog dijela instrukcije odrediti adresu početka potprograma; Staviti adresu u programsko

brojilo;

} Inače

Obaviti instrukciju na način određen dekodiranim operacijskim kodom; } Dok je (procesor uključen);

7. Definirati osnovne pojmove: . program, proces, dretvu.

- Program - statični niz instrukcija, pohranjen na papiru, disketi, memoriji itd.
- Dretva - niz instrukcija koje se izvode i kontroliraju proces.
- Proces - program u izvođenju; skup računalnih resursa koji omoguduju izvođenje programa; okolina u kojoj se program izvodi; sve što je potrebno za izvođenje programa; sastoji se od:

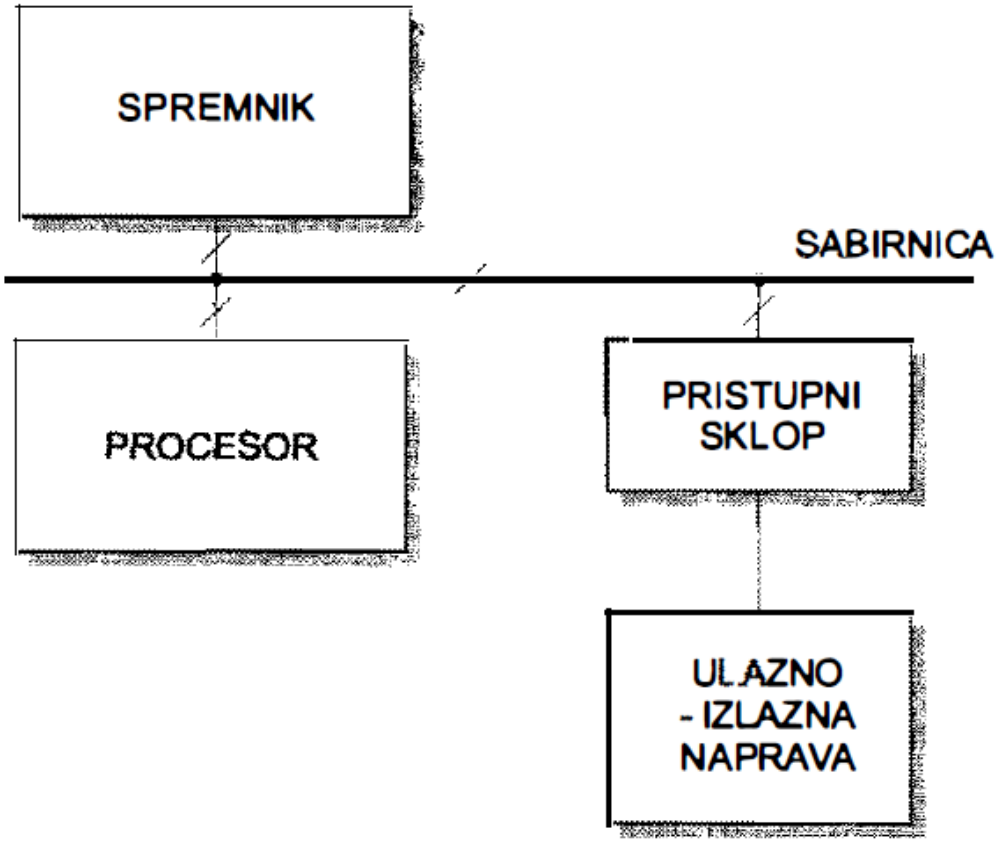
  - barem jedne dretve
  - zajedničkog adresnog prostora
  - adresnog prostora rezerviranog za svaku pojedinu dretvu
  - stog, kazaljke stoga, opisnici datoteka, opisnici cjevovoda, redovi poruka, semafori, uvjetne varijable,
    zaključavanja.

8. Kako je moguć višeprogramski rad na jednoprocesorskom računalu?

Tako da se svaka dretva izvodi naizmjence pa dobijemo privid istovremenosti. Ključna je pravilna izmjena konteksta dretve.
[Višedretveni se rad može provesti i u jednoprocesorskom računalu, i to tako da taj jedan jedini procesor naizmjence "provlači" jednu od više dretvi. U tom se slučaju ne može govoriti o ubrzavanju odvijanja procesa, ali se može postići da procesor izvodi jednu od dretvi za vrijeme dok druga mora zbog nekog razloga čekati. S obzirom na to da svaka dretva za izvođenje treba "svoj procesor", u jednoprocesorskom sustavu mora se, pri prebacivanju izvođenja s jedne dretve na drugu, osigurati da svaka dretva radi sa svojim skupom registara. To se može postići tako da se sadržaj registara procesora one dretve čije se izvođenje želi prekinuti pohrani na neko rezervirano mjesto u spremniku, a u registre procesora smjesti sadržaje koji pripadaju dretvi čije izvođenje upravo treba započeti .Ako se takva promjena sadržaja registara obavi ispravno, onda se stvara privid da svaka dretva posjeduje "vlastiti" procesor. ***Sadržaj registara procesora zovemo kontekstom dretve, a promjenu sadržaja registara promjenom konteksta]***

3.

3.1 . Skicirati način spajan ja UI naprave na sabirnicu.



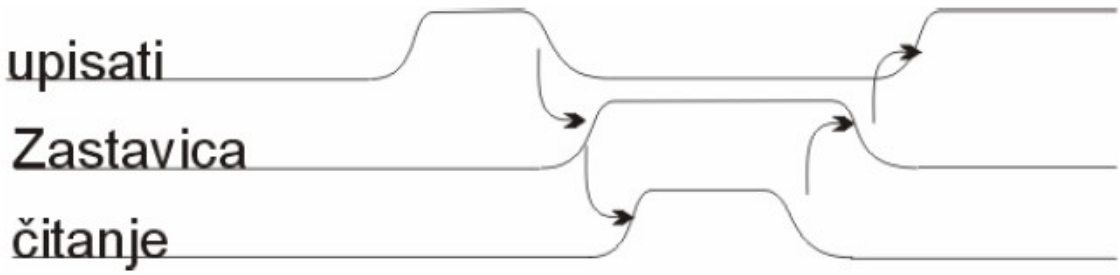
2. Što je radno čekanje?

Režim rada procesora u kojem procesor čeka na određeni događaj u programu (npr. pojava zastavice) i troši vrijeme dok se taj događaj ne dogodi.

RADNO ČEKANJE je prijenos znakova gdje:

1. procesor treba pročitati registar stanja
2. dok je odgovarajuća zastavica u registru stanja ZASTAVICA=0 nemoj raditi ništa nego pročitaj ponovo RS (registar stanja), tek kad je ZASTAVICA=1 može se pročitati podatkovni registar (PR).

3. Skicirati signale dvožičnog rukovanja.



4. Što se zbiva kada se dogodi prekid?

Pojava prekidnog signala prebacuje procesor u tzv. sustavski način rada. S obzirom na to da će se na taj način pozivati potprogrami koji čine jezgru operacijskog sustava taj se

način rada procesora naziva još i jezgrenim načinom rada (engl. system mode - sustavski način, kernel mode - jezgreni način) . Dretva koja je pojavom prekidnog signala prekinuta najčešće izvodi neki korisnički posao i zbog toga kažemo da se ona obavlja u korisničkom načinu rada (engl. user mode) .
U našem modelu procesor pri prelasku u jezgreni način djeluje tako da:

- privremeno onemogući daljnje prekidanje;
- djeluje na adresni dio sabirnice tako da adresira odvojeni dio spremnika (time je spremnik podijeljen na dva dijela: korisnički dio i sustavski dio) ;
- aktivira drugi sustavski registar kazaljke stoga, koji se uz već spomenuti korisnički registar kazaljke stoga nalazi u procesoru (time se omogućuje ostvarenje posebnog sustavskog stoga, koji nema veza s korisničkim stogom, taj se stog mora nalaziti u jezgrenom adresnom prostoru);
- pohranjuje trenutačni sadržaj programskog brojila na sustavski stog;

5. Kako treba nadopuniti ponašanje procesora da on omogućuje prekidni rad bez sklopa za prihvat prekida?

```
ponavljati {
        .
        .
        .
//na kraju dolazi
      ako je (prekidni signal postavljen) {
                        zabrani prekidanje;
                        aktivirati sustavski adresni prostor i sustavsku kazaljku stoga;
                        PC ↔ sustavski stog;
                        PC ↔ adresa početka podprograma;
                        }
      } sve dok je (procesor uključen);
```

ili

Pojavio se prekidni signal, zabranjeno je prekidanje, i programsko brojilo nalazi se na sustavskom stogu;
{
Pohrani kontekst;
Ustanovi uzrok prekida, odnosno odredi indeks prekida i;
Ako je $(1 < i < n)$ *{*
Postavi oznaku čekanja K_Z[i]=1 ;
Poništi zastavicu u registru stanja prekida i;
Dok je $((postoji\ K\_Z[j] != 0) \wedge (J > T\_P))${
Odabrati najveći j;
K_Z[j] = 0;
Pohraniti kontekst sa sustavskog stoga i T_P u KON[j];
T_P = j;
Omogući prekidanje;
Prijeđi u korisnički način rada;
Pozovi potprogram za obradu prekida j;

```
                    Zabrani prekidanje;
                    Prijeđi u sustavski način rada;
                    Vrati sadržaj iz KON[j] na sustavski stog i u varijablu T_P;
                }
            }
            Obnovi kontekst sa sustavskog stoga;
            Omogući prekidanje;
            Vrati se iz prekidnog načina rada;
    }
```

4.

1. Koje računalne resurse dijele dretve istog procesa?

Dretve koje djeluje unutar jednog procesa dijele sva sredstva koja je operacijski sustav stavio na raspolaganje tom procesu, a posebno na adresni prostor procesa. Dretve koje djeluju unutar jednog procesa mogu neposredno pristupiti do svih adresa adresnog prostora svog procesa.

2.Što je zajedničko dretvama različitih procesa?

4. Navesti uvjet nezavisnosti zadataka.

$$(X_i \cap Y_j) \cup (X_j \cap Y_i) \cup (Y_i \cap Y_j) = \emptyset.$$

5. Navesti uvjete koje mora zadovoljavati algoritam međusobnog isključivanja dretvi.

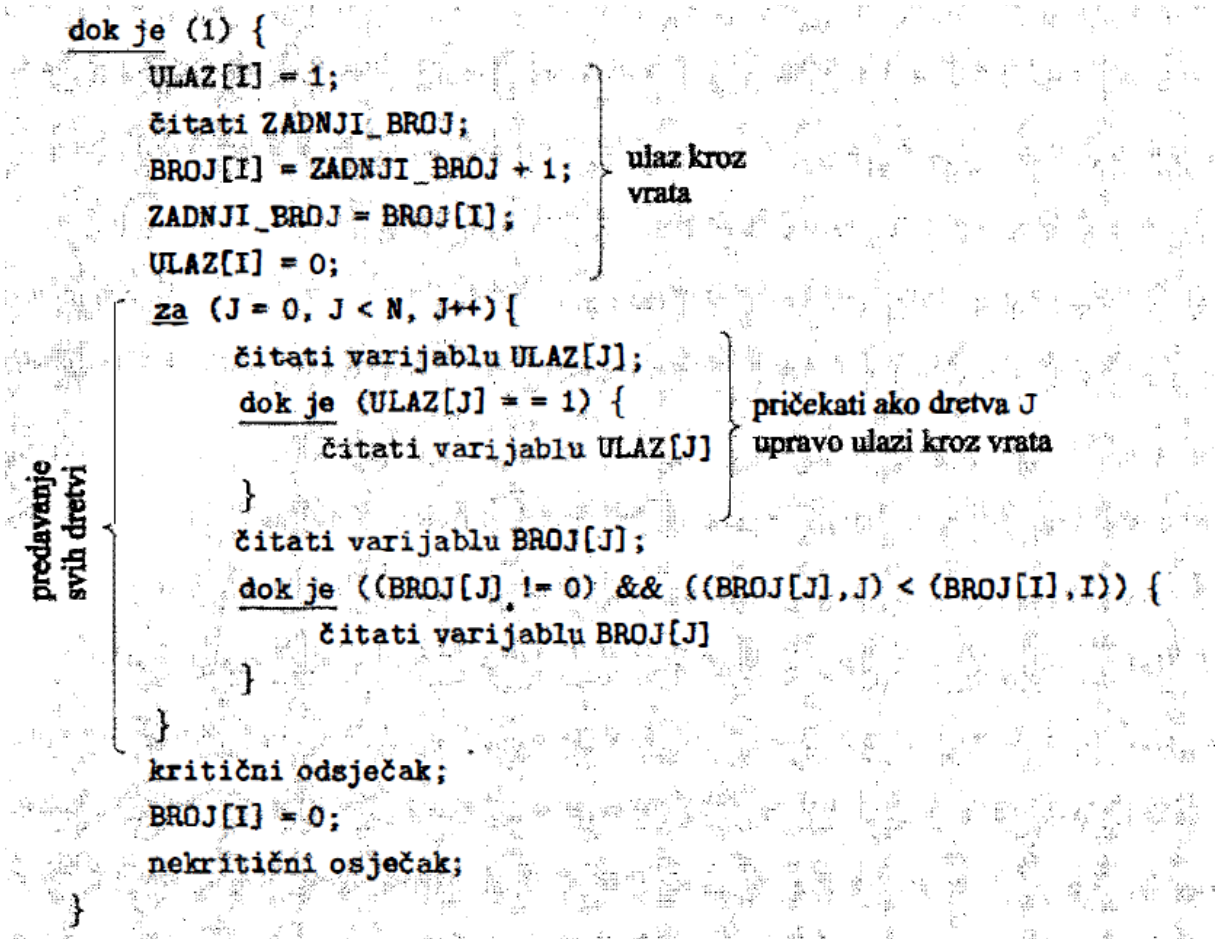
- Samo jedna dretva u nekom trenutku može biti u k.o.(kritičnom odsječku)
- Algoritam međusobnog isključivanja mora djelovati i onda kada su brzine izvođenja dretvi različite
- Ako neka dretva zastane u nekritičnom odsječku N.K.O. to ne smije spriječiti drugu dretvu da uđe u K.O.
- Izbor koja dretva treba ući u K.O. mora se obaviti u konačnom vremenu

8. Navesti Dekkerov, odn. Lamportov algoritam.

- Dekkerov algoritam

```
dok je (1) {
        ZASTAVICA[ I ]=1;
        čitati varijablu ZASTAVICA[ J ];
                dok je (ZASTAVICA[ J ]!=0) {
                        čitati varijablu PRAVO;
                        ako je (PRAVO!=I) {
                                ZASTAVICA[ I ]=0;
                                dok je (PRAVO!=I) {
                                        čitati varijablu PRAVO;
                                }
                                ZASTAVICA[ I ]=1;
                        }
                        čitati varijablu ZASTAVICA[J];
                }
                K.O.;
                PRAVO=J;
                ZASTAVICA[ I ]=0;
                NK.O.;
        }
}
```

Lamportov algoritam:



5.

1 . Što predstavlja pojam "ulazak u jezgru" i kada se zbiva?

- Poziv jezgrine funkcije - ulazak u jezgru - zbiva se pod utjecajem sklopovskog ili programskog prekida.
- Pojavom sklopovskog prekida ili programski izazvanog prekida događa se sljedeće:
- onemogućava se prekidanje;
- programsko brojilo se stavlja na sustavski stog
- prekidni podsustav premješta kontekst iz registra procesora u sustavski stog;
- prekidni podsustav poziva odgovarajuću jezgrinu funkciju
- Jezgrina funkcija koja je prekidom pozvana na svom početku morala bi najprije premjestiti kontekst sa sustavskog stoga u opisnik dretve Aktivna_D. To se opisuje instrukcijom pohraniti kontekst u opisnik Aktivna_D.

2. Na što se svodi "izlazak iz jezgre"?

- Izlazak iz jezgre svodi se na aktiviranje jedne od dretvi, tj. dretve koja se nalazi na prvom mjestu u redu Pripravna_D.

3. Navesti izvore prekida u jednostavnom modelu jezgre.

u sustavu ćemo imati tri vrste prekida:

- sklopovske prekide od ulazno-izlaznih naprava;
- periodne sklopovske prekide od sata;
- programske prekide koje izazivaju dretve.

4. Od čega se sastoji jezgra operacijskog sustava?

Jezgra se sastoji od svoje strukture podataka i funkcija smještenih u sustavski dio spremnika. U strukturi podataka jezgre moraju se pohraniti sve informacije o dretvama potrebne za donošenje odluka o njihovu izvođenju.

5. Navesti sadržaj opisnika dretve.

. Uz podatke važne za opis stanja dretve u opisniku su predviđena i mjesta za smještanje kazaljki koje će nam omogućiti da ga lako premještamo iz jedne dinamičke strukture (liste) u drugu.), u opisniku su predviđena mjesta za smještanje sljedećih parametara:

- ***Identifikator_procesa*** označava kojem procesu pripada dretva (prirodni broj).
- ***Identifikator_dretve*** omogućuje međusobno razlikovanje dretvi (prirodni broj).
- ***Stanje_ dretve*** je parametar koji označava u kojem se stanju dretva trenutačno na lazi. (Vidjet ćemo da dretva u našem modelu može biti pasivna, aktivna, blokirana i
  pripravna za izvođenje.)
- ***Prioritet_ dretve*** je parametar koji određuje prednost dretvi pri dodjeljivanju pro cesora (prioritet ćemo izraziti prirodnim brojem, i to tako da veći broj označava veći prioritet).
- ***Početna _ adresa_ dretvenog_ adresnog_ prostora*** i ***Veličina_ prostora*** opisuju smještanje dretvenog adresnog prostora unutar procesnog adresnog prostora.
- ***Adresa_prve _ instrukcije*** sadrži adresu na kojoj je smještena adresa prve instrukcije dretve.
- ***Zadano _ kašnjenje*** je parametar koji će odrediti odgađanje izvođenja dretve za za
  dani broj perioda otkucaja sata.
- ***Prostor _ za_ smještanje _ konteksta*** poslužit će nam za smještanje konteksta dretve onda kada dretva bude prekinuta u svom izvođenju

6. Navesti strukture podataka jezgre.

- Liste:

1. Pasivne_D - kada se dretva nalazi na samo jednoj listi (postojeće_D), onda je u pasivnom stanju
2. Aktivna_D - dretve koje se izvode; broj članova u toj listi jednak je broju procesora.
3. Pripravne_D - ako se ne izvode, a spremne su. Prema načinu formiranja red može biti - po redu prispjeća ili prioritetni
4. Red BSEM[i] - dretve koje čekaju na binarnom semaforu
5. Red OSEM[j] - dretve koje čekaju na općem semaforu
6. Red odgođen1_D - zadano kašnjenje dretve
7. Red UI[k] - ima ih koliko ima U/I naprava

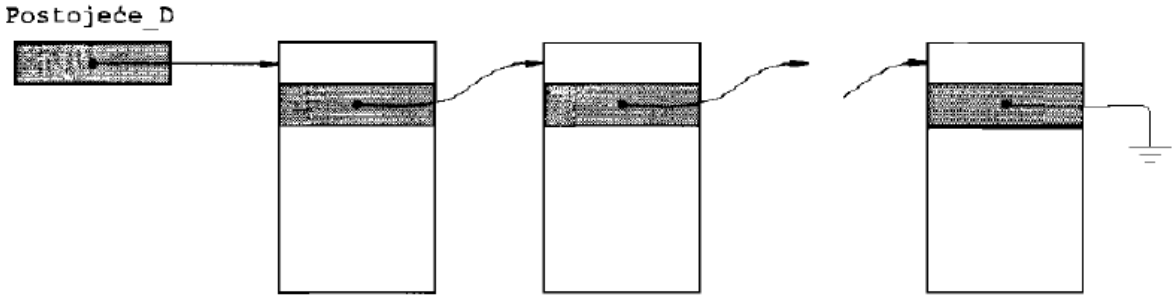
- Opisnici dretvi

7. Koja su blokirana stanja dretvi?

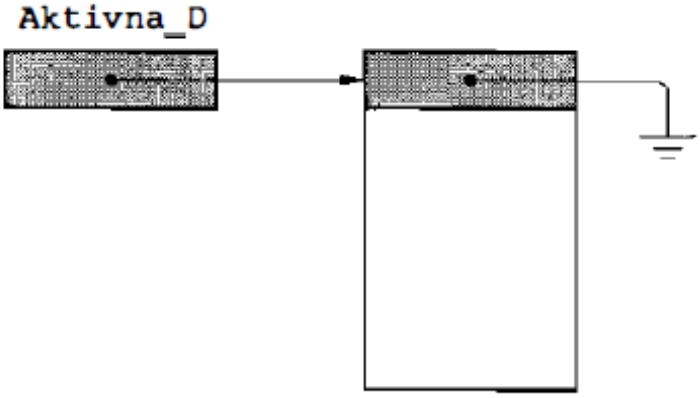
- Čekanje na binarnom semaforu
- Čekanje na općem semaforu
- Čekanje na istek zadanog intervala kašnjenja (odgođene dretve)
- Čekanje na završetak U/I operacije.

8. Skicirati graf mogućih stanja dretvi.

PASIVNO STANJE

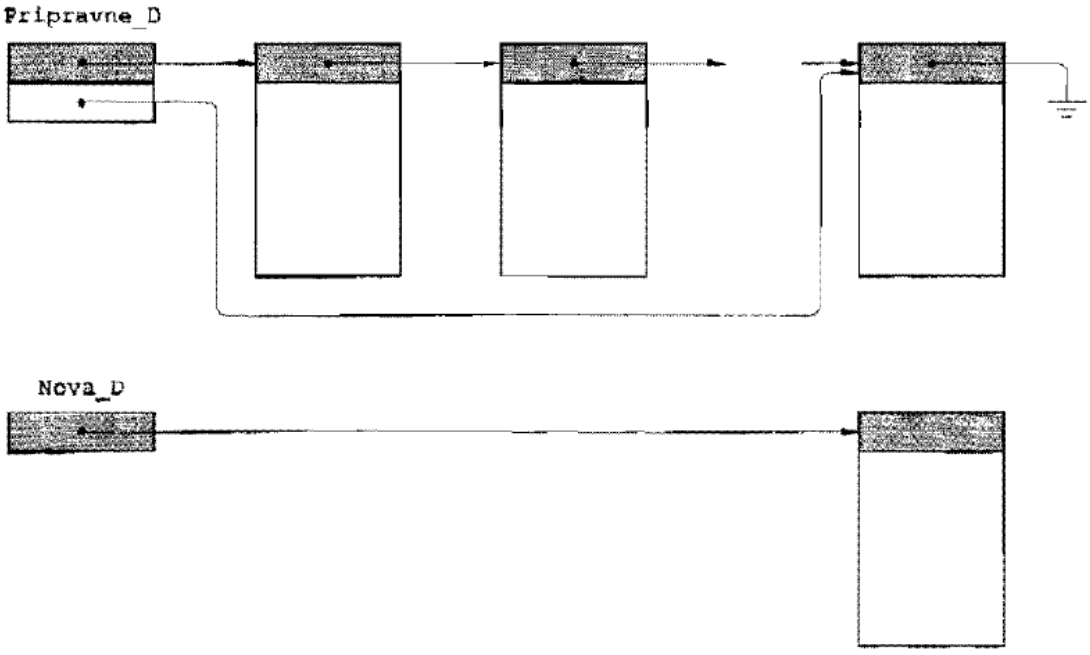


AKTIVNO STANJE

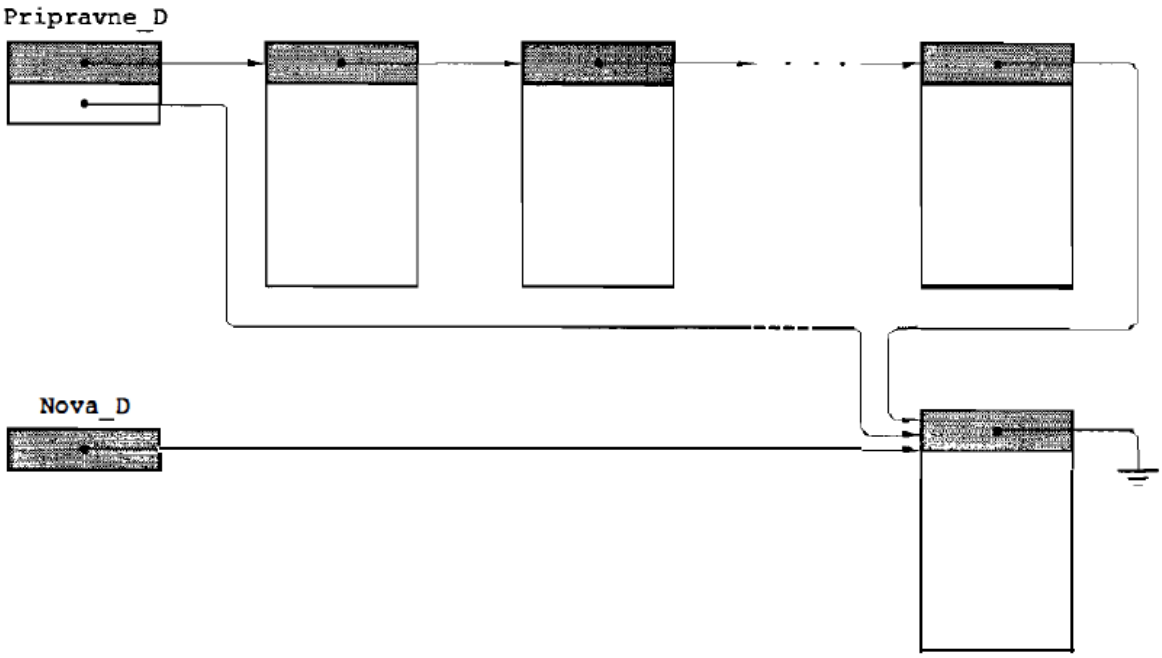


PRIPRAVNO STANJE DRETVI PO REDU PRISPJEĆA

a) **prije umetanja opisnika nove dretve**

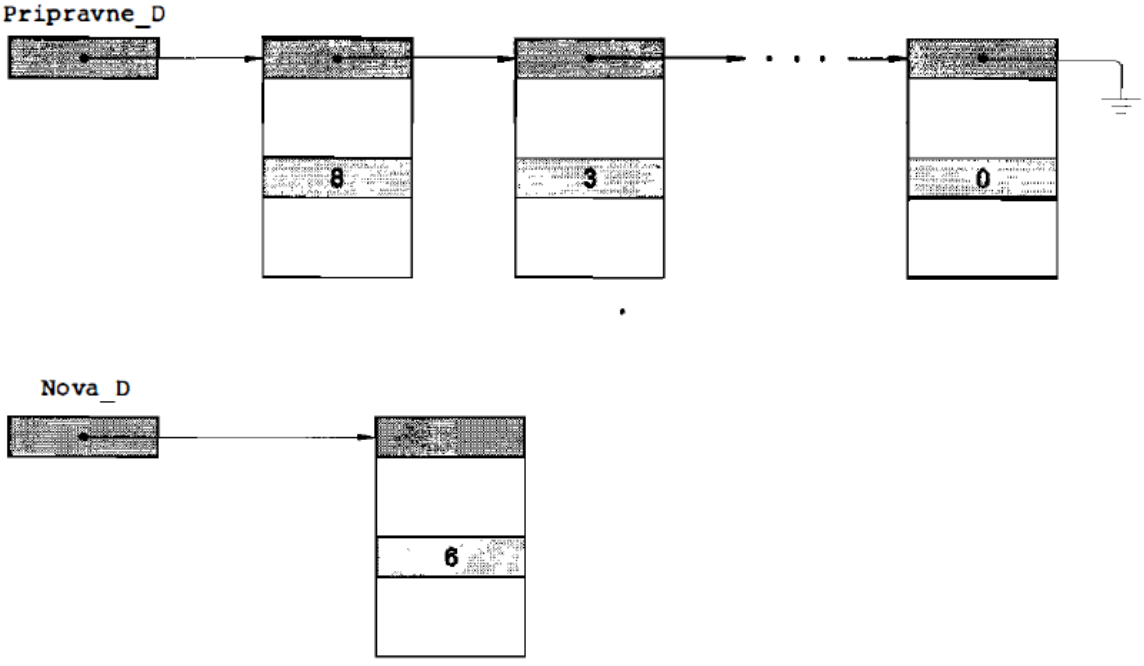


b) **nakon umetanja opisnika nove dretve**

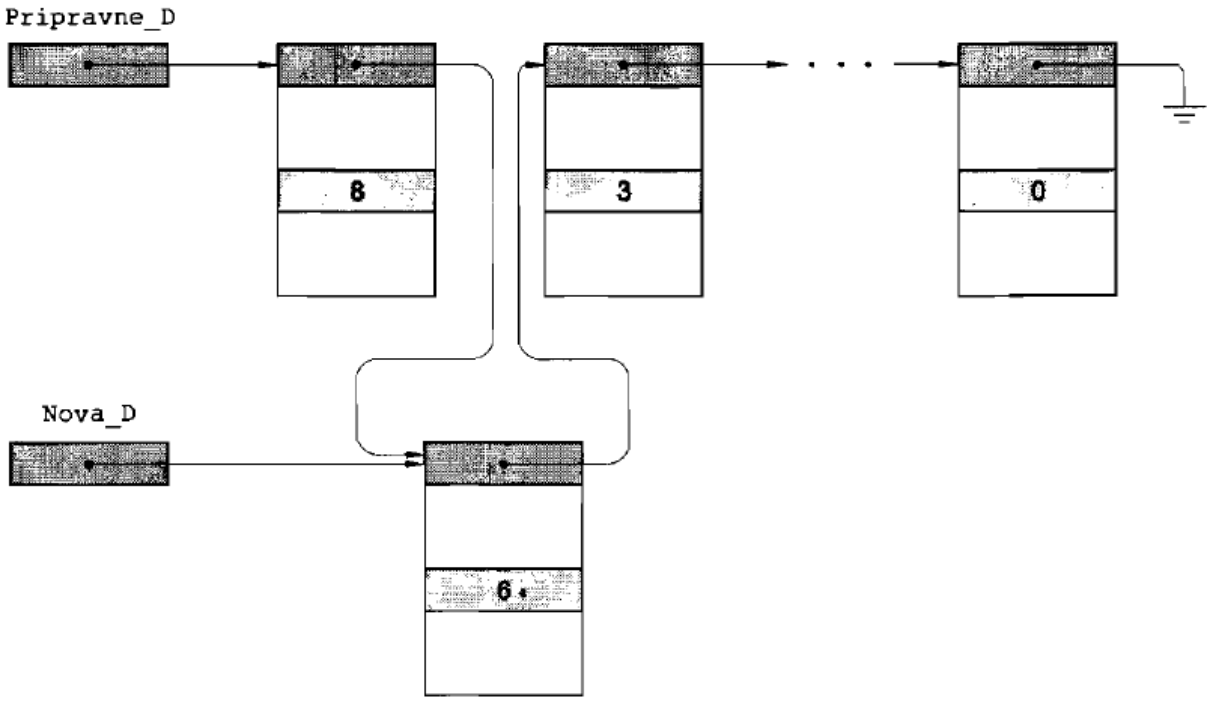


PRIPRAVNO STANJE DRETVI PO PRIORITETIMA

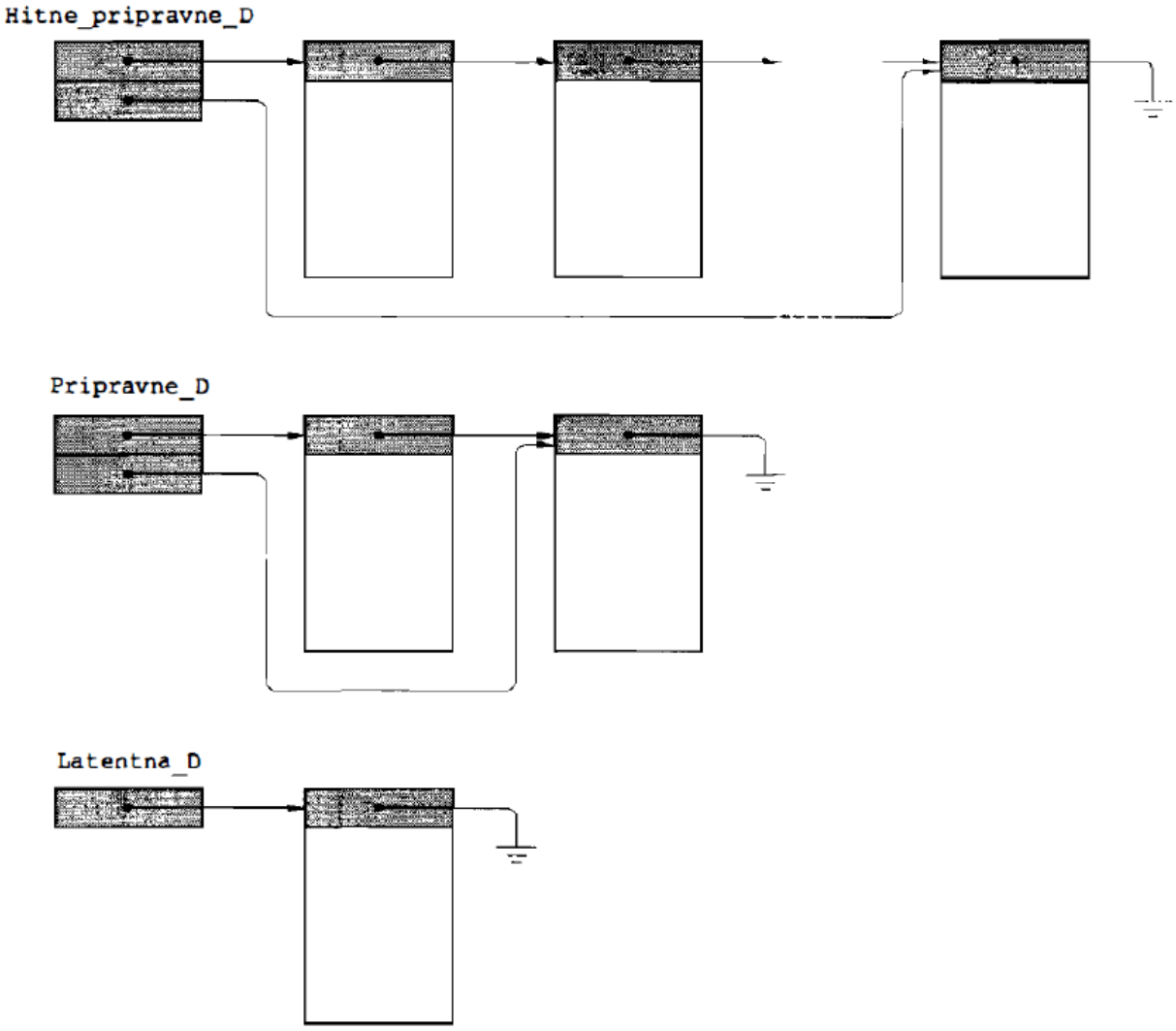
a) **prije umetanja opisnika nove dretve**



b) **nakon umetanja opisnika nove dretve**



PRIPRAVNE DRETVE S DVIJE RAZINE PRIORITETA I LATENTNOM DRETVOM



9. Što obavlja instrukcija "aktivirati prvu dretvu iz reda Pripravne_ D"?

- Postupak aktiviranja se opisuje sljedećim slijedom instrukcija:
premjestiti prvi opisnik iz reda Pripravne_D u red Aktivna_D;
obnoviti kontekst iz opisnika Aktivna_D;
omogućiti prekidanje;
vratiti se iz prekidnog načina; // vraća u procesor sadržaj PC-a i prevodi procesor iz sustavkog u korisnički način rada.
- To opisujemo instrukcijom aktivirati prvu dretvu iz reda Pripravna_D

11. Čemu služe jezgrini mehanizmi binarni i opći semafor (BSEM i OS)?

Binarni semafor jezgrin je mehanizam koji omogućuje ostvarenje međusobnog isključivanja dretvi. Binarni semafor sastoji se od jedne varijable, imenujmo je Bsem [I].v , koja igra ulogu zastavice i pridružene joj kazaljke koja pokazuje na red opisnika dretvi koje nisu uspjele proći semafor. Vrijednost semafora Bsem [I] . v == 1 označava da je semafor prolazan, a vrijednost Bsem [I] . v == O označava da je semafor neprolazan. U sustavu možemo imati i više semafora.

Opći semafor razlikuje se od binarnog semafora po tome što njegova vrijednost Osem [J] . v može poprimiti vrijednost cijelog broja, a ne samo vrijednosti O ili 1.Kada dretva pokušava proći opći semafor, jezgra smanji njegovu vrijednost za jedan i ako je nakon toga
Osem[J].v >=O, dopušta dretvi prolaz. Ako je Osem [J] . v < O, onda se dretva blokira u

redu pridruženome tom općem semaforu. Neka druga dretva koja zahtijeva postavljanje općeg semafora uzrokovat će povećanje vrijednosti Osem[J] . v. Time se može ispuniti uvjet za pokretanje dretve koja je bila blokirana.

6.

1. Sinkronizirati proizvođača i potrošača korištenjem brojačkog semafora.

```
Proizvođač:          Potrošač:
dok je(1){           dok je(1){
proizvesti           Ispitati_Osem(
poruku P;            1);
Ispitati_Osem(2      R=MS[IZ];
);
MS[UL]=P;            IZ=(IZ+1)%N;
UL=(UL+1)%N;         Postavi_Osem(
                     2);
Postavi_Osem(1       Potrošiti
);                   poruku R;
}                    }
```

Objašnjene varijabli: MS=međuspremnik

UL, IZ=kazaljke za kretanje po međuspremniku P, R=poruka

N=veličina međuspremnika

2. Sinkronizirati više proizvođača i više potrošača uz pomoć binarnih i brojačkih semafora.

```
Proizvođač:                          Potrošač:
Dok je (1){                          Dok je (1){
Proizvesti poruku P;                 Ispitati_Osem(i);
Ispitati_Osem(S);                    Ispitati_Bsem(K);
Ispitati_Bsem(P);                    Preuzeti poruku R iz prvog spremnika
                                     u
Dobaviti pretinac sa stoga           Red_Poruka[i];
SKLADISTE;
Postaviti P u pretinac i uvrstiti    Vratiti ispažnjeni pretinac na stog
ga u                                 SKLADISTE;
Red_Poruka[i];                       Postaviti_Bsem[K];
Postaviti_Bsem[P];                   Postaviti_Osem(S);
Postaviti_Osem(i);                   Potrošiti poruku R;
}                                    }
```

3. Sinkronizirati rad dviju dretvi tako da se one obavljaju naizmjenično.

```
Dretva Di:      Dretva Dj:
```

```
dok je (1){
Ispitati_Osem(i);
nešto raditi;
Postaviti_Osem(j);
}
```

```
dok je (1){
Ispitati_Osem(j);
nešto raditi;
Postaviti_Osem(i);
}
```

4. Što je potpuni zastoj?
Potpuni zastoj je stanje sustava u kojem su sve dretve blokirane u nekom redu uvjeta

5. Navesti nužne uvjete za nastajanje potpunog zastoja.

- neko sredstvo u istome trenutku može upotrebljavati samo jedna od dretvi (međusobno isključivo)
- dretvi se sredstvo ne može oduzeti - ona ga otpušta sama kada ga više ne treba
- dretva drži dodijeljeno joj sredstvo dok čeka na dodjelu dodatnog sredstva.

6. Što je monitor?
Monitor je nadzorni program koji nadzire nakupine funkcija za razrješavanje nekih cjelovitih problema suradnje dretvi (suradnji procesa) - Monitor je jezgrin mehanizam za sinkronizaciju.

8.
1.Gdje se generiraju adrese unutar procesora?
Sve adrese se generiraju unutar procesora (iznimno se adrese generiraju u pristupnim sklopovima s neposrednim pristupom spremniku). U načelu adrese koje se generiraju unutar procesora jesu:

- adrese instrukcija koje dolaze iz programskog brojila,
- stogovne adrese koje dolaze iz registra kazaljke stoga i

- adrese operanada i rezultata operacija (adrese podataka) koje se stvaraju na temelju sadržaja adresnih dijelova instrukcija.

2. Kako je podijeljen procesni adresni prostor?
Dretveni prostori + zajednički prostor

3. Opisati organizaciju smještaja sadržaja na magnetskom disku (cilindri, staze, sektori).
- Tvrdi disk može imati više ploča, a sve ploče učvršćene su na istu osovinu i vrte se konstantnom brzinom. U smjeru radijusa ploča pomiču se glave za čitanje i pisanje
- Glava se pozicionira iznad staze, a sve staze jednakih polumjera svih diskova leže na zamišljenom plaštu valjka kojeg nazivamo *cilindar*.
- Svaka staza podijeljena je na jednake dijelove koje nazivamo *sektorima*
- Na disku se ne adresiraju pojedinačni bajtovi, nego blokovi bajtova. Blokovi bajtova sastoje se iz jednog ili više sektora.. Upravljački sklop diskovne jedinke mora omogućiti prijenos blokova.

4. Čime je određena jedinstvena adresa svakog sektora na disku?

Svaki sektor ima jedinstvenu adresu (sastavljena od rednog broja ploče, rednog broja staze na ploči i redni broj sektora)

5. Koliko iznosi ukupno trajanje prijenosa podataka tvrdi disk - radni spremnik?

Vrijeme prijenosa podataka ovisi o brzini prijenosa koja je određena brzinom okretanja i gustoćom bitova na stazi (tipična brzina: od 1 do 5 MB/s)..

6. Od čega se sastoji trajanje traženja staze (seek time)?

- ubrzavanja glave
- kretanje konstantnom brzinom (za velike razmake staza)
- usporavanja ruke
- finog pozicioniranja

7. Koliko iznosi prosječno vrijeme traženja u odnosu na vrijeme koje je potrebno za prijelaz preko svih staza?

Uobičajeno se za vrijeme traženja uzima vrijeme potrebno za prijelaz preko trećine ukupnog broja staza

8. Zbog čega nastaje rotacijsko kašnjenje i koliko ono iznosi?

rotacijsko kašnjenje (eng. rotation latency) - to je vrijeme koje je potrebno da sektor dođe ispod glave (određeno je brzinom okretanja ploče). Nastaje zbog toga što se nakon postavljanja glave na stazu mora prije početka prijenosa, pričekati da se ispod glave pojavi traženi sektor. U najlošijem slučaju može se desiti da je traženi sektor upravo prolazio ispod glave kad je ona prispjela na stazu pa se tada treba čekati puni okretaj

- Prosječno rotacijsko kašnjenje jednako je polovini trajanja jednog okretaja diska.

12. Opisati postupke statičkog i dinamičkog dodjeljivanja spremnika.

Statičko dodjeljivanje spremnika

- U ranijim sustavima se cijeli spremnik dijelio na dijelove stalne veličine, tzv. *stalne particije* tako da je:
- svaki od programa je pripreman za cijelu particiju
- particije su uglavnom bile različitih veličina
- Programi pripremljeni za jednu particiju nisu se jednostavno mogli premjestiti na drugu particiju jer je to zahtijevalo novo generiranje apsolutnog oblika programa. Otuda i naziv *statičko raspoređivanje* - programi se tijekom svog boravka u sustavu nalaze uvijek u istom dijelu radnog spremnika.

Dinamičko dodjeljivanje spremnika

Programi se upisuju jedan iza drugog. Dodan je sklop s kojim se početna adresa pribraja svakoj adresi koju proces proizvede. Programi se priređuju tako da svaki ima svoj adresni prostor. Važno je da se prije izvođenja u bazni registar pohrani aktualna početna adresa

13. Navesti vrste fragmentacije prilikom statičkog dodjeljivanja spremnika.

Unutarnja fragmentacija: Programi nisu jednake velicine kao particije pa ce djelovi particija ostati neiskorišteni. Vanjska fragmentacija: Tijekom rada se može dogoditi da svi procesi čiji su programi smješteni u istu particiju bivaju blokirani pa ta particija radnog spremnika ostaje prazna. Pritom može postojati više procesa čiji programi čekaju na dodjelu radnog spremnika, ali oni ne mogu biti napunjeni u radni spremnik jer nisu pripremljeni za tu particiju

14. Problem fragmentacije prilikom dinamičkog dodjeljivanja spremnika ne može se izbjeći, ali se može ublažiti. Kako?

- Tijekom rada neki procesi završavaju, a drugi traže spremnik. Stoga mogu nastati nepopunjeni dijelovi ili "rupe". Te rupe mogu postati male, pa se ne može pronaći program koji bi je mogao popuniti. To je tzv. *fragmentacija spremnika*.

- Osnovni način suzbijanja fragmentacije je održavanje rupa što većima da bi se u njih mogli smjestiti novi programi. To se postiže tako da se :
- pri svakom oslobađanju nekog procesnog prostora novonastala rupa spaja s eventualnim susjednim rupama u novu veću rupu
- pri svakom novom zahtjevu za spremničkim prostorom potraži najmanja rupa u koju se može smjestiti novi program

15. Unatoč tomu što se problem fragmentacije prilikom dinamičkog dodjeljivanja spremnik može ublažiti, fragmentacija može postati prevelika. Što tada treba učiniti?

- Postoji još i mogućnost da se u nekom trenutku, kad fragmentacija bude prevelika, privremeno zaustavi izvođenje dretvi i "preslože" programi u kompaktniji prostor. To se naziva "*sakupljanje otpadaka*" (engl. garbage collection).

18. Kako je podijeljen logički, a kako fizički adresni prostor u sustavu sa straničenjem?

- Logički adresni prostor zamišljeno se dijeli na stranice koje imaju veličinu koja je cjelobrojna potencija broja 2, a adrese unutar stranica se kreću od 0 do 2^p -1.
- Stvarni (fizički) spremnik podijeljen je na *okvire* (eng.frame) koji su jednaki veličini stranice logičkog adresnog prostora. Kako bi se pri dodjeli fizičkog spremnika izbjegao problem fragmentacije, osnovna postavka straničenja svodi se na to da se stranice logičkog adresnog prostora mogu biti smještene u okvire fizičkog spremnika proizvoljnim redoslijedom.

19. O čemu ovisi veličina fizičkog i logičkog adresnog prostora?

Logički adresni prostor ovisi o arhitekturi, a fizički o RAM-u

20. Mogu li stranice logičkog adresnog prostora biti smještene u okvire fizičkog spremnika proizvoljnim redoslijedom?

Mogu. Time se izbjegava problem fragmentacije.

21 . Čemu služi tablica prevođenja? od kojih se elemenata sastoji?

- U posebnoj se tablici vodi evidencija o tome u kojem je okviru smještena pojedina stranica. Ona se koristi pri svakom pristupu fizičkom spremniku. Stoga, ona mora biti sastavni dio spremničkog međusklopa, jer se prevođenje logičke adrese u fizičku adresu mora obaviti što je moguće brže

Za svaku stranicu postoji zapis u tablici prevođenja:

-Redni broj okvira u kojem se nalazi stranica

-razne zastavice (korištena, globalna, prljava, zaštićena...)

22. U čemu se razlikuju prekidi izazvani zbog promašaja stranice kod straničenja na zahtjev od ostalih vrsta prekida?

Prekid zbog adresiranja nepostojeće stranice se dogodi usred instrukcije. Nakon obrade prekida mora se ponoviti instrukcija unutar koje se dogodio prekid. Kod ostalih vrsta prekida, postojanje prekida se provjerava nakon izvođenja instrukcije, a ponovno pokretanje dretve započinje sljedećom instrukcijom

24. Koju informaciju nosi bit čistoće? Gdje se on nalazi?

Bit čistoće „kaže“ je li stranica u okviru mijenjana u odnosu na njenu sliku na pomoćnom spremniku, nalazi se u opisniku okvira

25. Čemu služi posmačni registar povijesti?

Za izvedbu LRU strategije. Pri punjenju stranice u neki okvir inicijaliziraju se na 0 njezin bit pristupa i posmačni registar. Kada se pristupi stranici, bit pristupa stranice se postavlja u

1. Prekid od sata periodno posmakne sve bitove pristupa u posmačne registre i zatim bit pristupa izbriše. Sadržaj registra može se pročitati kao binarni broj. Manji broj pokazuje da se stranicu nije duže vrijeme koristilo. Kada se pojavi potreba za izbacivanjem stranice izbacuje se ona s najmanjim brojem

26. Opisati sljedeće strategije za izbacivanje stranica: FIFO, LRU, OPT te satni algoritam.

1.FIFO (*First In First Out*) - izbacuje stranicu koja je najdulje u radnom spremniku.
- Ova strategija je teško ostvariva, a osim toga pati od Beladyeve anomalije, tj. s povećanjem broja raspoloživih okvira može porasti broj promašaja.

2.OPT (*Optimalna Strategija*) - izbacuje stranicu koja se najdalje u budućnosti neće upotrebljavati
- Ova strategija nije ostvariva, za njeno provođenje treba se poznavati odvijanje cijelog procesa .

3.LRU (*Last Recently Used*) - izbacuje onu stranicu koja najdalje u prošlosti nije upotrebljavala

Satni algoritam (engl. clock algorithm). Sve stranice koje su smještene u okvire radnog spremnika razvrstavaju se u listu onim redom kako su ulazile u radni spremnik. Ta se lista može promatrati kao FIFO red i njome bi bilo moguće ostvariti FIFO strategiju. Međutim, lista se obilazi posebnom kazaljkom koja se s kraja liste vraća na njezin početak i tako se ostvaruje kružni obilazak liste (od tuda dolazi i naziv algoritma) i u neku ruku slijedi LRU strategija s dva razreda stranica. Kada se pojavi potreba za praznim okvirom, izbacuje se stranica na koju pokazuje kazaljka ako je njezin bit pristupa A jednak O i kazaljka se u listi pomiče za jedno mjesto. Stranica će se preskočiti ako je bit pristupa A jednak 1. Ako kazaljka kružno obiđe cijelu listu i pronalazi sve bitove pristupa jednake 1 , onda će početi izbacivati stranice u koje se pristupalo.

27. Opisati strategiju izbacivanja stranica u Intel x86 arhitekturi kada se u obzir uzimaju dvije zastavice za označavanje stanja stranice.

Zastavice koje se koriste su bit pristupa (A) i bit nečistode (D). Prije se izbacuju stranice u kojima se nije pristupalo i koje su čiste.

28. U kojim se stanjima mogu nalaziti pojedini okviri tijekom rada?

- Aktivno stajne - okvir je dodjeljen jednom od procesa i njegov se redni broj (kazaljka koja pokazuje na njegov početak) nalazi u tablici prevođenja tog procesa
- Slobodno stanje - okvir se nalazi u povezanoj listi oslobođenih okvira
- Slobodno stanje s obrisanim sadržajem - okvir je spreman za dodjeljivanje

9.

1 . Navesti sadržaj opisnika datoteke.

Sadrži indeksno dohvatljive kazaljke na nakupine sektora. Uz pretpostavku da je sektor velik 1024 B te da kazaljke imaju 32 bita, u jedan sektor može se smjestiti 256 kazaljki. U opisniku se nalazi 13 kazaljki. i to:
• 10 neposrednih kazaljki,
• l jednostruko indirektna kazaljka,
• l dvostruko indirektna kazaljka i
• 1 trostruko indirektna kazaljka.

- Naziv datoteke
- Tip
- Lozinka
- Ime vlasnika
- Prava pristupa

- Vrijeme stvaranja
- Vrijeme zadnje uporabe
- Ime posljednjeg korisnika, i slično
- Opis smještaja

2. Gdje su pohranjeni:
a) opis smještaja datoteke, - u opisniku datoteke
b) opisnik datoteke i - na magnetskskom disku
e) datotečna tablica - u opisniku datoteke

3. Navesti sadržaj datotečne tablice.
- broj sektora po disku (kapacitet)
- broj slobodnih sektora (veličina slobodnog prostora)
- informacije o slobodnim sektorima
- tablica opisnika pohranjenih datoteka

4. Na koji se način može prikazati slobodan prostor na disku?

- bitovni prikaz
- lista slobodnih sektora
- lista nakupina sektora

5. Opisati bitovni prikaz slobodnog prostora na disku.
U nizu bitova svakom sektoru odnosno nakupini sektora pripada jedan bit čija je vrijednost jednaka 1 kada je sektor odnosno nakupina zauzeta, inače 0.

6. Opisati prikaz slobodnog prostora u obliku liste slobodnih blokova.
Svaki element liste sadrži tri lokacije. Prvo je kazaljka na sljedeći slobodan element, drugo adresa prvog slobodnog sektora, a treće broj slobodnih sektora.

7. Opisati način smještaja datoteka u UNIX datotečnim podsustavima (i-nade).
Opisnik datoteke u UNIXu zove se i-node. Uz pretpostavku da je sektor velik 1KB u jedan sektor se može smjestiti 256 kazaljki. U opisniku se nalazi 13 kazaljki i to:

- 10 neposrednih kazaljki
- 1 jednostruko indirektna kazaljka
- 1 dvostruko indirektna kazaljka
- 1 trostruko indirektna kazaljka

Pristup do datoteke obavlja se na sljedeći način:

- Prvih deset sektora dohvaća se neposrednim kazaljkama čime se može dohvatiti sve sektore datoteka manjih od 10KB
- Jedanaesta kazaljka pokazuje na sektor u kojem se nalazi sljedećih 256 kazaljki (256 + 10)KB
- Dvanaesta kazaljka pokazuje na sektor u kojem su kazaljke na 256 sektora svaki s 256 kazaljki (256*256+256+10)KB
- Trinaesta kazaljka pokazuje na trostruko stablo kazaljki (256*256*256+256*256+256+10)KB = 16GB

8. Opisati način smještaja datoteka u NTFS datotečnom podsustavu. Što je MFT? Kako se pohranjuju "male" datoteke?

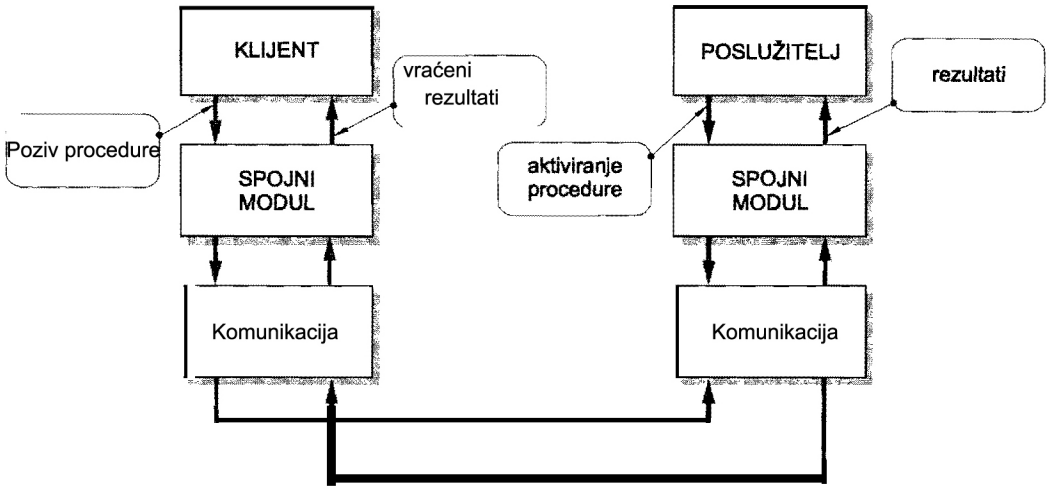
Diskovni prostor dodjeljuje se po skupinama sektora (po 1, 2, 4 ili 8 sektora). NTFS sadrži datoteku MFT (glavna tablica datoteka) u kojoj se nalaze opisnici svih datoteka (uključujući i nje same). Jedan MFT zapis ima veličinu jedne skupine sektora (npr. 4KB). Za veće datoteke zapis sadrži indekse skupina sektora i ako je potrebno dodatno proširenje opisa.
Datoteka se dijeli na dijelove koji su jednako veliki kao nakupine sektora (virtualne skupine). Za smještaj datoteka pronalazi se što više uzastopnih skupina sektora i dodjeljuje se datoteci (ako nema uzastopnih dodjeljuju se pojedinačne skupine).
Male datoteke smještaju se unutar MFT zapisa.

10.

1. Navesti mehanizme za međusobnu komunikaciju između procesa na jednom računalu.
   U načelu su moguća dva načina komuniciranja:
   - zajednička uporaba izvornih podataka u dijeljenom spremničkom prostoru;
   - uporaba kopija podataka zasnovana na razmjeni poruka.

2. Kako mogu komunicirati procesi na udaljenim računalima?

3. Opisati mehanizam poziva udaljenih procedura. (RPC).



Mehanizam poziva udaljenih procedura ostvaruje se iznad TCP razine tako da poziv udaljene
procedure izgleda kao uobičajeni poziv potprograma. Taj poziv prihvaća spojni modul (engl. stub) koji oblikuje poruku i prepušta poruku komunikacijskom sustavu. Poruka se upućuje računalu koje će izvesti proceduru. U njegovu spojnom modulu u poruci će se prepoznati ime procedure i njezine ulazne vrijednosti. Procedura nakon svog završetka predaje rezultate svom spojnom modulu, koji će oblikovati povratnu poruku i predati j e komunikacijskom sustavu. Na strani koja je pozvala proceduru spojni modul rezultate prispjele povratnom porukom vraća u proces koji je pozvao proceduru. Proces koji poziva proceduru zapravo je klijent, a proces koji izvodi proceduru je poslužitelj.

4. Navesti mehanizme koji omogućuju međusobno isključivanje dretvi i procesa na jednom računalu.

5. Opisati centralizirani protokol međusobnog isključivanja u raspodijeljenim sustavima.

U potpuno centraliziranom protokolu jedan od čvorova u mreži može se proglasiti odgovornim za ostvarenje međusobnog isključivanja. U njemu se nalaze svi podaci i ostali čvorovi moraju od tog čvora tražiti dozvolu za ulazak u kritični odsječak. Proces čvora Pi (jedna dretva unutar tog procesa) koji želi ući u kritični odsječak mora poslati poruku zahtjev (i) centralnom čvoru. Centralni čvor će prihvatiti zahtjev i poslati poruku odgovor (i) čvoru Pi kada ustanovi da on smije ući u kritični odsječak. Proces Pi (odnosno njegova dretva) mora pričekati da dobije odgovor prije nego što uđe u kritični odsječak. Nakon što dretva u čvoru i obavi kritični odsječak ona šalje poruku izlaz (i) centralnom čvoru. Centralni čvor organizira red prispjelih zahtjeva i dopušta ulazak u kritični odsječak po redu prispijeća nakon što primi poruku izlaz (i) . Za ostvarenje ulaska u kritični odsječak za jedan proces potrebne su, dakle, samo dvije poruke ! Na taj se protokol može gledati i na drugi način. Može se reći da centralni čvor posjeduje jedan objekt (značku) koji će slati čvorovima koji traže ulazak u kritični odsječak. Značka se može poslati samo jednomu od čvorova. Čvor mora vratiti značku u konačnom vremenu centralnom čvoru. Osnovni nedostatak ovoga protokola jest velika ovisnost o centralnom čvoru. Ako se centralni čvor pokvari, protokol će u potpunosti zatajiti.

6. Opisati protokol s plutajućom značkom.

Pokazuje se da nam centralni čvor nije ni potreban. U raspodijeljenom se sustavu definira
jedna značka koja kao poruka ciklički putuje kroz sve čvorove.
Kada poruka značke prispije u čvor i proces P i će:
• zadržati značku ako želi ući u kritični odsječak te poslati značku sljedećem čvoru tek nakon što ga završi ili
• proslijediti značku sljedećem čvoru ako ne želi ulaziti u kritični odsječak.

Ovaj ciklički protokol po svojoj je zamisli sličan mehanizmu dodjele sabirnice u višeprocesorskom čvrsto povezanom sustavu. Tamo se sabirnica redom ciklički nudi svim procesorima, a pravo pristupa koriste samo oni procesori koji to žele.
Nedostaci ovakvog protokola uključuju neotpornost na ispad bilo kojeg čvora koji će prekinuti lanac prosljeđivanja značke. Nadalje, i u trenucima kada niti jedan čvor ne želi ući u kritični odsječak, značka se i dalje prosljeđuje kroz čvorove, nepotrebno opterećujući komunikacijski sustav.

7. Opisati lokalni i globalni logički sat.

Lokalni logički sat
U svakom čvoru i može postojati lokalni logički sat Ci koji se ponaša kao brojilo. On povećava svoju vrijednost nakon svakog karakterističnog događaja unutar procesa P i '
Unutar jednog procesa može se jednoznačno utvrditi redoslijed događaja.
Nekom događaju a pripada vrijednost logičkog sata Ci Ca) , a događaju b vrijednost logičkog
sata Ci Cb) .
Ako je Ci (a) < Ci (b) , onda se događaj a dogodio prije događaja b, što možemo pisati:
a → b,
gdje → označava relaciju "dogodilo se prije".
Relaciju "dogodilo se prije" možemo jednoznačno odrediti:
• za dva događaja unutar jednog procesa ili
• za događaj slanja poruke od strane jednog procesa i primitka te iste poruke u drugom procesu.
Prema tome, relacija a → b vrijedi:

• ako su a i b događaji unutar jednog procesa i događaj a se zbio prije događaja b ili
• ako je a događaj slanja poruka od strane jednog procesa i b događaj primitka te iste poruke od strane drugog procesa.
Nadalje, relacija a → b je tranzitivna pa vrijedi:
(a → b) 1\ (b → e) povlači da je (a → e) .
Ako ni (a → b) ni (b → a) nije istinito, onda događaji a i b nisu vremenski uređeni.

Globalni logički sat
Na razini sustava može se uspostaviti skup pravila vremenskog uređenja koji nazivamo globalni logički sat. Globalni sat zasniva se na sljedećim pravilima:
• proces Pi povećava svoj logički sat Ci između svaka svoja dva događaja;
• kada proces Pi šalje poruku m, on uz nju pridodaje vremensku oznaku Tm = Ci ;
• kada proces P j primi poruku, on postavlja u svoj logički sat C j vrijednost koja je veća od njegove prethodne vrijednosti i veća od prispjele vremenske oznake Tm .
pri ovakvom ostvarenju globalnog sata može se dogoditi:
• da dva lokalna sata Ci i Cj imaju jednake vrijednosti i
• da se njihove poruke koje si međusobno šalju presretnu na putu noseći jednake vrijednosti
vremenskih oznaka.
U tom slučaju dobit ćemo jednake vremenske oznake za dva različita događaja. Po uzoru na Lamportov protokol opisan u odjeljku 4.4. uređenje možemo postići usporedbom vrijednosti indeksa koji su unaprijed pridjeljeni svakom čvoru. Prema tome, u sustavu se može podržavati globalni sat koji čine nakupine lokalnih satova.
Relacija vremenskog uređaja:
a => b (a "prethodi" b)
za događaj a iz procesa P i i događaj b iz procesa P j zadovoljena je kada je:
(Ci (a) < Cj (b))
ili:
(Ci (a) = Cj (b)) 1\ (i < j) .
Utvrđivanje redoslijeda događaja olakšat će nam ostvarenje mehanizama međusobnog isključivanja u raspodijeljenim sustavima.

8. Opisati raspodijeljeni Lamportov protokol.
Lamportov raspodijeljeni protokol zasniva se na uvažavanju vremenskog uređenja temeljenog na globalnom logičkom satu.
Kada proces Pi želi ući u kritični odsječak, generirat će poruku zahtjev (i , T (i)) i poslat će je svim ostalim procesima, gdje je T (i) jednaka vrijednosti logičkog sata Ci u trenutku slanja poruke.
Unutar svakog procesa nalazi se red poruka u kojem se svi zahtjevi za ulazak u kritični odsječak svrstavaju u skladu s relacijom " => " (prethodi) vremenskog uređenja u sustavu.
Protokol se izvodi u skladu s pet pravila koja se moraju ostvariti kao lokalne operacije pojedinih čvorova:
a) Kada proces Pi zahtjeva ulazak u kritični odsječak on:
• stavlja poruku zahtjev (i > T (i)) u svoj vlastiti red čekanja i
• šalje tu poruku svim ostalim procesima.
b) Kada proces Pj primi poruku zahtjev (i , T (i)) , on:
• uskladi svoj lokalni sat Cj u skladu s pravilima uspostave globalnog sata,
• stavlja u svoj red čekanja poruku zahtjev (i , T (i)) te
• šalje poruku odgovor (i , T (j)) procesu Pi, gdje je T (j) jednako novoj vrijednosti logičkog sata C j .
c) Proces Pi smije ući u kritični odsječak:
• kada se njegov vlastiti zahtjev nalazi na početku reda i

• kada je proces Pi primio poruke odgovora svih ostalih procesa.
d) Proces Pi obavlja izlazak iz kritičnog odsječka tako da:
• ukloni iz svog reda svoj zahtjev (i , T (i)) i
• pošalje svim ostalim procesima poruku izlazak (i , T (i)) , gdje je T (i) jednaka vrijednosti iz prvotno poslanog zahtjeva.
e) Kada proces Pj primi poruku izlazak (i , T (i)) , on iz svog reda čekanja uklanja zahtjev procesa Pi .
Za ulazak i izlazak iz kritičnog odsječka u sustavu se razmijeni 3 . (N - 1) poruka, i to:
(N - 1) poruka zahtjev,
(N - 1) poruka odgovor i
(N - 1) poruka izlazak.

9. Opisati protokol Ricarda i Agrawala.

Ricart i Agrawala pokazali su da se ukupni broj poruka u sustavu s N čvorova može svesti
na 2 . (N - 1) poruku. To se pojednostavljenje protokola može postići tako da procesi šalju odgovore na primljene poruke samo onda ako ustanove da ne žele ulaziti u kritični odsječak ili ustanove da proces koji je postavio zahtjev ima pravo prvenstva.
Protokol se ostvaruje sljedećim pravilima:
a) Kada proces Pi zahtjeva ulazak u kritični odsječak, on:
• šalje poruku zahtjev (i . T (i» svim ostalim procesima (gdje je T (i) trenutačna vrijednost lokalnog sata u čvoru i) .
b) Kada proces Pj primi poruku zahtjev (i , T (i)) , on:
• uskladi svoj lokalni sat Cj u skladu s pravilima uspostave globalnog sata;
• šalje poruku odgovor (j , T (i)) , ako ne želi ulaziti u kritični odsječak ili ako je zahtjev procesa Pj za ulazak došao kasnije;
• proces Pj , dakle, neće poslati odgovor ako postoji njegov zahtjev čija vremenska oznaka (j , T (j)) prethodi vremenskoj oznaci (i , T (i)) .
c) Kada proces Pi primi odgovore svih ostalih čvorova, on smije ući u kritični odsječak.
d) Kada proces Pi izlazi iz kritičnog odsječka, on će poslati poruku odgovor (j , T (i)) svim procesima čiji zahtjevi kod njega čekaju na odgovor.
Za ulazak i izlazak iz kritičnog odsječka u sustavu se razmijeni 2 . (N - 1) poruka, i to:
(N 1) poruka zahtjev i
(N - 1) poruka odgovor.

11. POGLAVLJE

1 . OBJASNITI POJMOVE IDENTIFIKACIJA, AUTENTIFIKACIJA I AUTORIZACIJA.

Računalni sustav mora provesti postupak provjere identifikacije. Taj se postupak naziva **autentifikacijom** l (engl. authentication) . Autentifikacija se provodi prilikom prijave za rad na sustavu (engl. login) . Pritom se mora utvrditi autentičnost samo jedne strane, tj. korisnika. Korisnik, naime, ne mora provjeravati identitet računala. Međutim, u raspodijeljenim sustavima autentifikacija se često mora provoditi dvostrano jer obje strane u komunikaciji moraju dokazivati svoj identitet. **Identifikacija** može biti i složeniji proces od procesa autentifikacije ukoliko se pretražuje cijela baza korisnika. Primjerice, otisak prsta treba se usporediti sa svima u bazi podataka kako bi se identificirala osoba koja je dala samo otisak prsta, a nije se i predstavila. S druge strane, u postupku se

autentifikacije samo provjerava ispravnost lozinke : nakon predstavljanja se otisak prsta uspoređuje samo s jednim podatkom u bazi. Mehanizmi dopuštanja pristupa (engl. Access control) pojedinim sredstvima nazivaju se autorizacijom pristupa (engl. authorization)

identifikacija = predstavljanje

autentifikacija = identifikacija + verifikacija

autorizacija = autentifikacija + provjera ovlasti, tj. provjera prava pristupa

2. NAVESTI VRSTE NAPADA NA SIGURNOST RAČUNALNOG SUSTAVA.

1. Prisluškivanje
3. Promjena sadržaja poruka
4. Izmišljanje poruka
5. Lažno predstavljanje
6. Poricanje (engl. repudiation)

3. NAVESTI SIGURNOSNE ZAHTJEVE.

Povjerljivost ili tajnost, Raspoloživost, Besprijekornost, Autentičnost, Autorizacija, Neporecivost

4. NAVESTI SVOJSTVA SIMETRIČNOG I ASIMETRIČNOG KRIPTOSUSTAVA

Simetrični kriptosustavi koriste isti ključ K i za kriptiranje i za dekriptiranje. Oni se, u načelu, zasnivaju na uporabi logičke operacije isključivo lU (engl. exclusive OR ili kraće: XOR). Označimo li dvije moguće vrijednosti logičke varijable s Oi 1, operacija XOR dat će kao rezultat vrijednost 1 onda kada samo jedna od ulaznih varijabli ima vrijednost

Asimetrični kriptosustavi imaju različite ključeve kriptiranja KE i dekriptiranja KD i mogu se opisati već navedenim izrazima:

C = E(P, KE)'

P =D(C, KD),

P = D(E(P, KE), KD).

5. NA KOJI NAČIN SE KRIPTIRA JASNI TEKST M KORISTEĆI JEDNOKRATNU BILJEŽNICU (ONE TIME PAD)?

Jedan od mogućih načina kriptiranja bio bi, dakle, taj da se kao ključ upotrijebi neki dogovoreni tekst s odgovarajućim brojem znakova, tj . odgovarajućom duljinom ključa. Takav kriptosustav naziva se jednokratnom bilježnicom (engl. One Time Pad).

Ako razgovijetni tekst koji želimo kriptirati ima više znakova, onda bi se on morao podijeliti

na blokove koji su jednaki duljini ključa (1) i kriptirati svaki blok posebno.

6. NAVESTI NEKOLIKO SIMETRIČNIH KRIPTOSUSTAVA.

1. Data Encryption Standard (DES)
2. Utrostručeni DES, 3 DES
3. Izbijeljeni DES, DESX
4. Kriptosustav I DEA
5. Napredni kriptosustav AES

7. KOJE SU MOGUĆE DULJINE KLJUČA U KRIPTOSUSTAVIMA DES, IDEA I AES?

DES – 56 bita simetričnim algoritmom. Trostruki DES 192 bita

IDEA- 128 bitova

AES- veličina ključa od 128, 192 i 256 bitova.

8. **NAVESTI POSTUPAK KRIPTIRANJA I DEKRIPTIRANJA UTROSTRUČENIM DES KRIPTOSUSTAVOM.**

Razbijanje DES kriptiranja može se otežati uporabom višestrukog DES kriptiranja. Umjesto jednog ključa K upotrebljavaju se tri ključa: KI , K2 i K3 (ključ K je ukupne duljine od 1 68 bita). Ovakav način trostrukog kriptiranja omogućuje da se 3 DES može upotrijebiti kao DES ako se sva tri ključa odaberu tako da budu jednaka. Postoji i inačica 3 DES koja koristi duljinu ključa od 1 12 bita (prvi i zadnji ključ su jednaki).

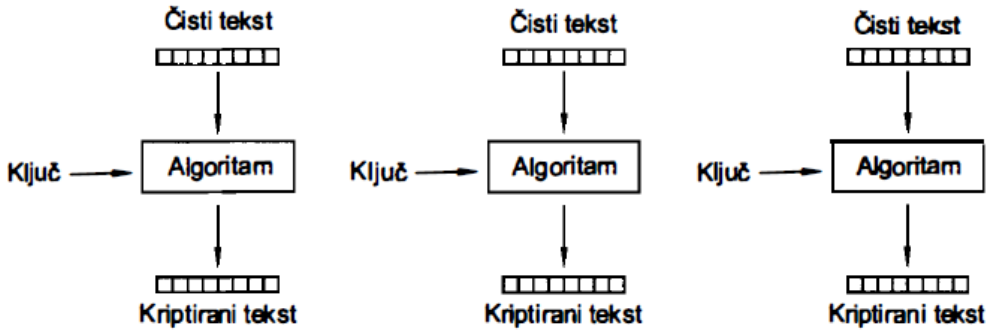
9. **NAVESTI POSTUPAK KRIPTIRANJA I DEKRIPTIRANJA IZBIJELJENIM DES KRIPTOSUSTAVOM.**

Jedan od načina za otežavanje razbijanja DES kriptosustava tzv. je bijeljenje teksta (eng.whitening). Uz 56-bitovni ključ Kl kriptiranja upotrebljavaju se još dva 64-bitovna ključa K2 i K3 koji "izbjeljuju" blokove tekstova. Prije kriptiranja blokovi razgovijetnog teksta duljine 64 bita podvrgavaju se XOR operaciji s ključem K2 , a nakon DES kriptiranja obavlja se dodatna XOR operacija s ključem K3 .

10. **ČEMU SLUŽE SUPSTITUCIJSKE (S) TABLICE U KRIPTOSUSTAVIMA DES I AES?**

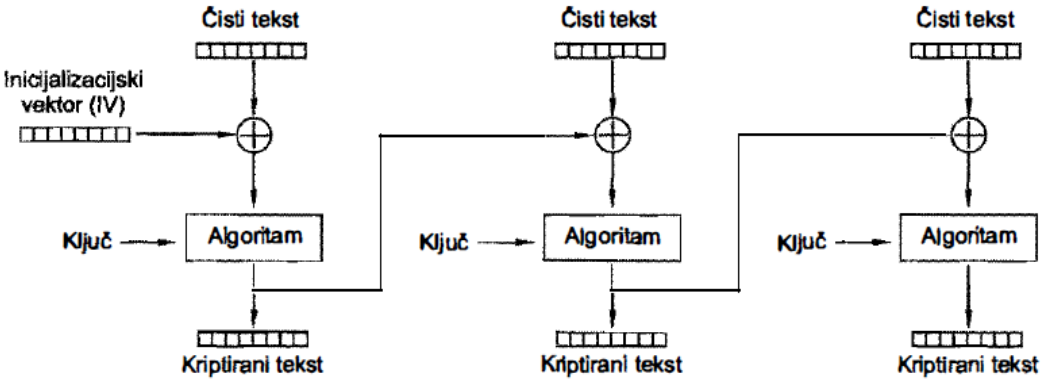
Ulaz u S tablicu je veličine 6 bita, a izlaz 4 bita. DES - Od ulaznih 48 bita dobivamo 32 bita nakon supstitucije.

11 . Skicirati ECB, CBC, CFB, OPB i CTR načine kriptiranja.



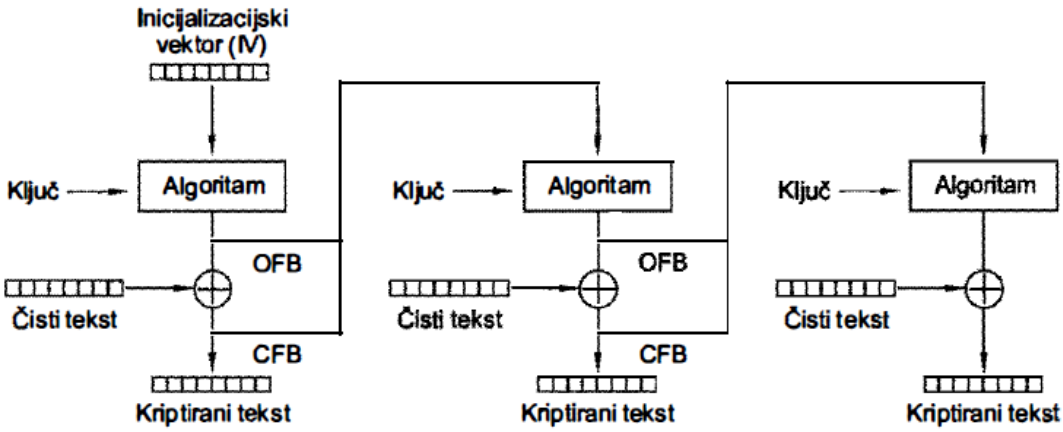
Slika 11.15. Kriptiranje u ECB načinu

CBC:



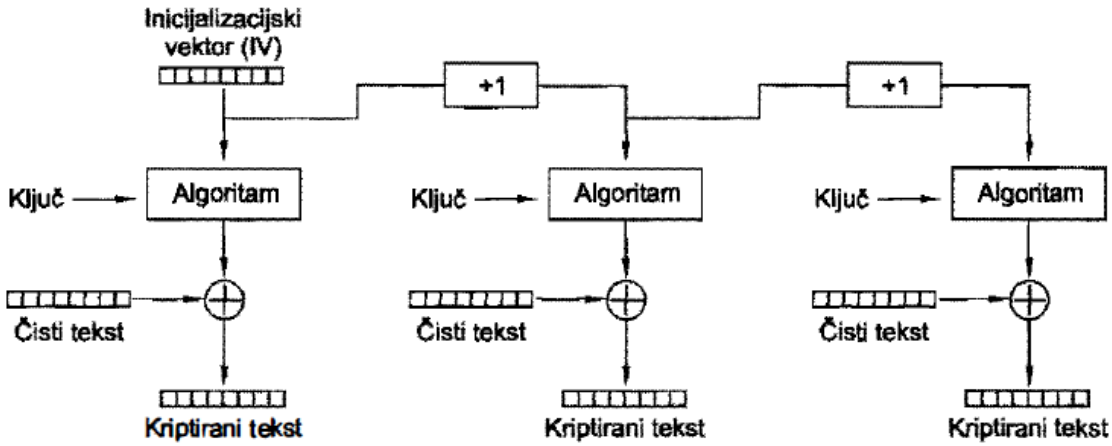
Slika 11.17. Kriptiranje ulančavanjem

CFB i OFB



Slika 11.19. CFB i OFB načini kriptiranja

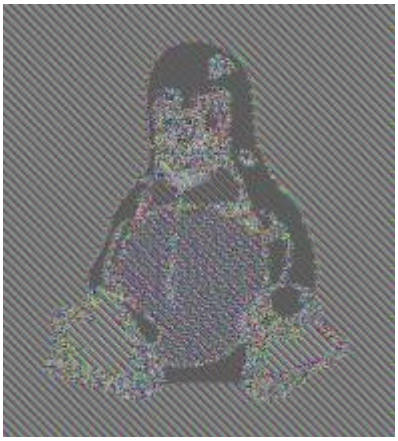
CTR



Slika 11.20. CTR način kriptiranja

12. KOJI JE OSNOVNI NEDOSTATAK ECR NAČINA KRIPTIRANJA?

Ne skriva dovoljno dobro uzorke podataka. Ne može djelotvorno sakriti sadržaj slike u bitmap formatu ako se primjenjuje najjednostavniji ECB način kriptiranja.



13. KOJI SU NAČINI KRIPTIRANJA POGODNI ZA KRIPTIRANJE TOKA PODATAKA?

CFB i OFB

14. **NAVESTI POSTUPAK GENERIRANJA KLJUČEVA U RSA KRIPTOSUSTAVU.**

1. Odabiru se dva velika prosta broja p i q (p >10100, q >10100).
2. Izračunava se umnožak n=p*q.
3. Izračunava se umnožak fi(n)=(p-1)*(q-1).
4. Odabire se broj d < fi(n) koji je relativno prost u odnosu na fi(n), tj. nema zajedničkih faktora sa fi(n).

5. Izračunava se broj e < fi(n) tako da bude:

e*d=k*fi(n)+1;

6. Par KE=(e,n) obznanjuje se i proglašava javnim ključem (engl. public key).

7. Par KD=(d,n) se taji i postaje privatni ključ (engl. private key).

15. **NAVESTI POSTUPAK KRIPTIRANJA KORISTEĆI RSA KRIPTOSUSTAV.**

- Kriptiranje se obavlja funkcijom kriptiranja

C = E(P, KE) = RSA (P, KE) = Pe mod n,

16. **ŠTO JE DIGITALNA OMOTNICA?**

Digitalna omotnica praktičan je način prenošenja kriptiranih podataka koji kombinira prednosti simetričnih i asimetričnih kriptosustava. Kako su asimetrični kriptosustavi presloženi i prespori za učinkovito kriptiranje većih količina podataka, podaci se kriptiraju simetričnim ključem te se taj simetrični ključ kriptira asimetričnim javnim ključem primatelja i dodaje kriptiranoj poruci. Time je riješen i problem prenošenja simetričnog tajnog ključa. Digitalna omotnica osigurava tajnost, ali ne i integritet poruke. Digitalna omotnica = {E(poruka, K); E(K, KEB)} Gdje je K tajni ključ simetričnog algoritma, a KEB javni ključ primatelja.

1 7. **ŠTO JE DIGITALNI POTPIS?**

Pošiljatelj iz poruke, koristeći jednu od funkcija za izračunavanje sažetka poruke (hash), izračunava sažetak te taj sažetak potom kriptira svojim privatnim ključem i dodaje izvornoj poruci. Digitalni potpis je sigurnosni mehanizam koji ne osigurava tajnost poruke, ali koristi se za ostvarenje autentičnosti, integriteta i neporecivosti [1]. Digitalni potpis = {poruka; E(Qp, KDA)} Gdje je Qp sažetak poruke a KDA tajni ključ pošiljatelja

1 8. **ŠTO JE DIGITALNI PEČAT?**

Digitalni pečat digitalno je potpisana digitalna omotnica. Zadovoljava četiri sigurnosna zahtjeva – tajnost, autentičnost, integritet i neporecivost [1]. Digitalni pečat = {digitalna omotnica; E(Qp, KDA)}

18. Što je digitalni pečat?

Digitalni pečat je potpisana poruka u digitalnoj omotnici, odnosno zapečaćena digitalna omotnica.

19. Koje sigurnosne zahtjeve osigurava digitalna omotnica, koje digitalni potpis, a koje digitalni pečat?

-Digitalnom omotnicom se postiže ispunjenje zahtjeva tajnosti slanja poruke.

-Digitalnim potpisom se utvrđuje besprijekornost poslane poruke.
-Digitalnim pečatom dobivamo digitalno potpisanu poruku u digitalnoj omotnici.

20. Što se ispituje uz pomoć heurističkog ispitivanja po Miller-Rabinu? Koji je rezultat ispitivanja?
Miller-Rabinovo ispitivanje je način ispitivanja brojeva sa ciljem determinacije da li je broj prost ili nije.

21. Navesti nekoliko funkcija za izračunavanje sažetka poruke.
-MD5
-SHA

22.Nabrojiti svojstva koja mora zadovoljiti funkcija za izračunavanje sažetka poruke.
-Otpornost na izračunavanje originala
-Otpornost na izračunavanje poruke koja daje isti sažetak
-Otpornost na kolizije

23. Pojasniti što znači da je neka funkcija sažimanja otporna na kolizije/izračunavanjeoriginala/izračunavanje poruke koja daje isti sažetak?
-otpornost na kolizije - ne smije biti moguće pronaći dvije različite poruke M1 i M2 za koje se dobiva isti sažetak.
-otpornost na izračunavanje originala - nemogućnost izračunavanja originalnog teksta poruke
-otpornost na izračunavanje poruke koja daje isti sažetak - nemogućnost pronalaska dviju različitih poruka koje daju isti sažetak

24. Kolika je veličina sažetka ako se koristi MD5/SHA-1 postupak?
-MD5 proizvodi sažetak duljine 128 bitova.
-SHA-1 proizvodi sažetak duljine 160 bitova.

25. U čemu se razlikuju postupci sažimanja SHA-0 i SHA-1?
-nemogu pronaći odgovor

26. Opisati Diffie-Hellmanov postupak razmjene tajnog ključa.
JEBENO SULUDO DUGAČAK I BESMISLEN ODGOVOR

27. Opisati napad čovjek u sredini (man in the middle attack) kada se koristi Diffie-Hellmanovpostupak razmjene tajnog ključa.
JEBENO SULUDO DUGAČAK I BESMISLEN ODGOVOR

28. Na koji se način obavlja zaštita pristupanja pojedinim sredstvima (autorizacija) koristeći matricu pristupa?
Uz pomoć zaštitnih pravila koja moraju za svaki par subjekt-objekt odrediti pravo pristupa i način na koji se objekt smije upotrebljavati.

29. Matrica pristupa rijetko je popunjena i može se praktičnije prikazati s pomoću lista. Navesti ipotpisati te liste.
-lista prava pristupa objektu
-lista dozvola za pristup objektima

30. Od čega se sastoji čvor Kerberos poslužitelja?
od autentifikacijskog poslužitelja i poslužitelja za dodjelu dozvola

31. U čemu se razlikuje ulazna dozvola i dozvola za pristup poslužitelju?
-nemogu pronaći odgovor

32. Što je digitalni certifikat? Navesti osnovni sadržaj digitalnog certifikata.
Digitalni certifikat je svojevrsna digitalna osobna iskaznica - predstavlja sredstvo kojim se dokazuje identitet na Internetu.

33. Navesti dijelove PKI sustava.
PKI se sastoji od više međusobno povezanih objekata, aplikacija i servisa:
•alata za upravljanje i nadgledanje sustava
•registracijskog centra ili registratora (Registration Authority ili skraćeno RA) koji obavlja registraciju korisnika
•certifikacijskog centra (Certification Authority ili skraćeno CA) koji se brine za izdavanje i valjanost certifikata
•baze izdanih certifikata (najčešće se koristi LDAP imenički servis) i liste opozvanih certifikata (Certification Revocation List ili skraćeno CRL)
•izdavača opozvanih certifikata korisničkog certifikata korisničkih aplikacija, servera itd., koji koriste PKI autorizaciju.

34. Opisati postupak jednostrane autentifikacije uz pomoć certifikata.
Protokol autentifikacije sudionika A se sastoji od šest koraka. Protokol opisan u idućih šest točaka pretpostavlja da su oba sudionika – A i B prijavljena u istom certifikacijskom centru.
1.Sudionik A šalje u razgovijetnom obliku svoj identifikator sudioniku B u poruci
M1 = (IDA).
2.Po primitku M1 sudionik B generira slučajni broj N i šalje u razgovijetnom obliku sudioniku A poruku
M2 = (N).
3.Po primitku M2 sudionik A kriptira svojim privatnim ključem N i šalje poruku M3 = E (N, KDA).
4.Sudionik B po primitku M3 šalje certifikacijskom centru C u razgovijetnom obliku poruku M4 = (IDA, RB),
gdje je RB kod kojim sudionik B zahtijeva od CA certifikat sudionika s identifikatorom IDA.
5.Po primitku M4 poslužitelj certifikacijskog centra C:
•iz svoje tablice na temelju IDA pročita CERAC;
•s pomoću svojeg privatnog ključa KDC kriptira CERAC i šalje poruku M5 = E (CERAC, KDC).
6.Po primitku M5 sudionik B:
•javnim ključem poslužitelja dekritpira M5 i dobiva
CERAC= D (E (CERAC, KDC), KEC);
•iz CERACsaznaje IDA, KEA i E (H (IDA, KEA), KDC));
•izračunava H (IDA, KEA) i dobiveni rezultat uspoređuje s dekriptiranom vrijednošću D (E (H (IDA, KEA), KDC)), KEC) čime provjerava dobiveni KEA, čime je ustvari proveo operaciju utvrđivanja ključa sudionika A pomoću ključa certifikacijskog centra C;
•s dobivenim KEA dekriptira raniju poruku M3 i dobiva
N = D (E (N, KDA), KEA);
•dobiveni N uspoređuje s originalom, čime prihvaća ili odbacuje sudionika A

35. Navesti i opisati preporučene X.509 autentifikacijske protokole.
-protokol s jednom porukom
-protokol s dvije poruke
-protokol s tri poruke

36. Navesti vrste sigurnosnih stijena.
-stijene koje filtriraju komunikacijske pakete
-stijene koje djeluju kao prividni poslužitelji
-stijene koje djeluju kao stvarni poslužitelji