OPERATIVNI SISTEMI

Slajdovi su kreirani na osnovu knjige “Operativni sistemi, principi unutrašnje organizacije i dizajna, 7. izdanje“, William Stallings, CET, Beograd, 2013.

Upravljanje memorijom

- *„Nikome neće trebati više od 640 KB memorije za personalni računar.“*

  Bill Gates, 1981. (nepotvrđeno)

Memorija

- Da bi se podržalo multiprogramiranje
  - OS mora da čuva više procesa u glavnoj memoriji
  - Memorija se deli između više procesa
- Radna memorija je veliki linearni niz bajtova
  - Svaki bajt ima svoju adresu
- Izvršavanje instrukcije
  - Procesor preuzima instrukcije iz memorije na osnovu vrednosti brojača instrukcija
  - Instrukcija se dekodira i može da uzrokuje dobavljanje operanada iz memorije
  - Nakon izvršenja instrukcije nad operandima, može biti potrebno smestiti rezultat u memoriju

Potreba za upravljanjem memorijom

- Lokacija podataka i instrukcija procesa
  - Glavna memorija i registri procesora su jedina memorijska skladišta kojima procesor može direktno da pristupa
  - U trenutku izvršavanja instrukcije, instrukcija i podaci koje koristi moraju biti u jednom od ova dva skladišta (ili u procesorskoj keš memoriji)

Radna vs masovna memorija

Radna memorija	**Masovna memorija**
Manjeg kapaciteta	Većeg kapaciteta
Brža	Sporija
Skuplja	Jeftinija
Nije trajna	Trajna

Izvor: www.youtube.com

Radna vs masovna memorija

- Upravljanje memorijom podrazumeva prebacivanje blokova podataka između masovne i glavne memorije
- Optimizovati trenutke i količinu podataka koji se razmenjuju
  - memorijski U/I uređaji su spori
  - kapacitet radne memorije je značajno ograničen

Zahtevi za upravljanje memorijom

- Računarski hardver i OS moraju da podrže
  - Relokaciju
  - Deljenje
  - Zaštitu

Relokacija

- Procesi (ili delovi procesa) se po potrebi ubacuju/izbacuju iz glavne memorije
- Proces može biti zamenjen na disk (*swapping*) i biti vraćen na drugu lokaciju u glavnoj memoriji (relociranje)
- Ne može se unapred znati u kojem delu memorije će se proces nalaziti
- U toku izvršavanja proces može da menja lokaciju u glavnoj memoriji

Deljenje memorije

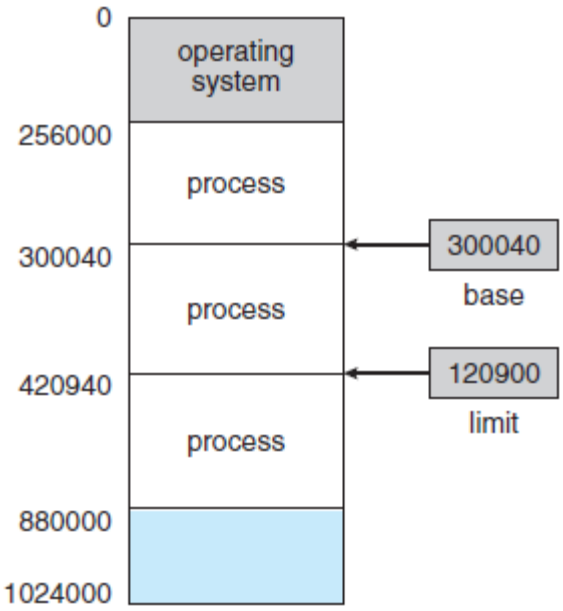
- Više procesa treba da pristupa istim delovima memorije
  - Npr. različite instance istog programa treba da dele programski kod
- Sistem za upravljanje memorijom treba da omogući kontrolisani pristup deljenim područjima u memoriji bez ugrožavanja zaštite

Zaštita

- Proces ne sme da pristupa bez dozvole lokacijama u koje su smešteni drugi procesi
- Zbog relokacije, lokacija procesa nije unapred određena i nepromenljiva
- Zato je nemoguće zaštitu sprovesti u vreme prevođenja (*compile time*)
- Zaštita mora biti sprovedena u toku izvršavanja (*run time*)
- Hardver procesora obezbeđuje zaštitu

Hardverska podrška za relokaciju i zaštitu

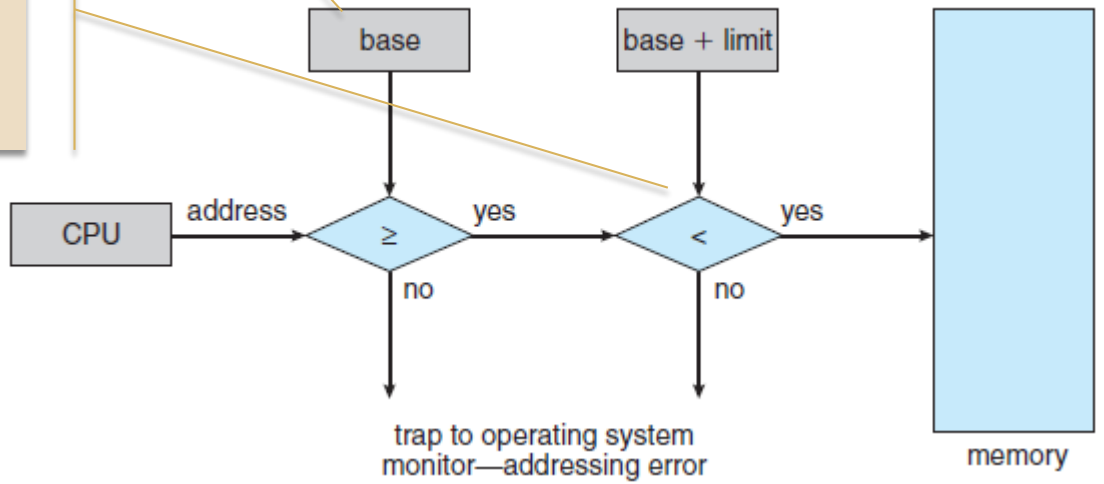
- Svaki proces mora da ima odvojen memorijski prostor
- Potrebno je definisati skup adresa kojima proces može da pristupa
- Dva procesorska registra
  - *base* – početna adresa na koju je proces smešten
  - *limit* – poslednja adresa (počevši od nula) kojoj proces pristupa



Hardverska podrška za relokaciju i zaštitu

Adresa početka bloka u kojem je proces trenutno

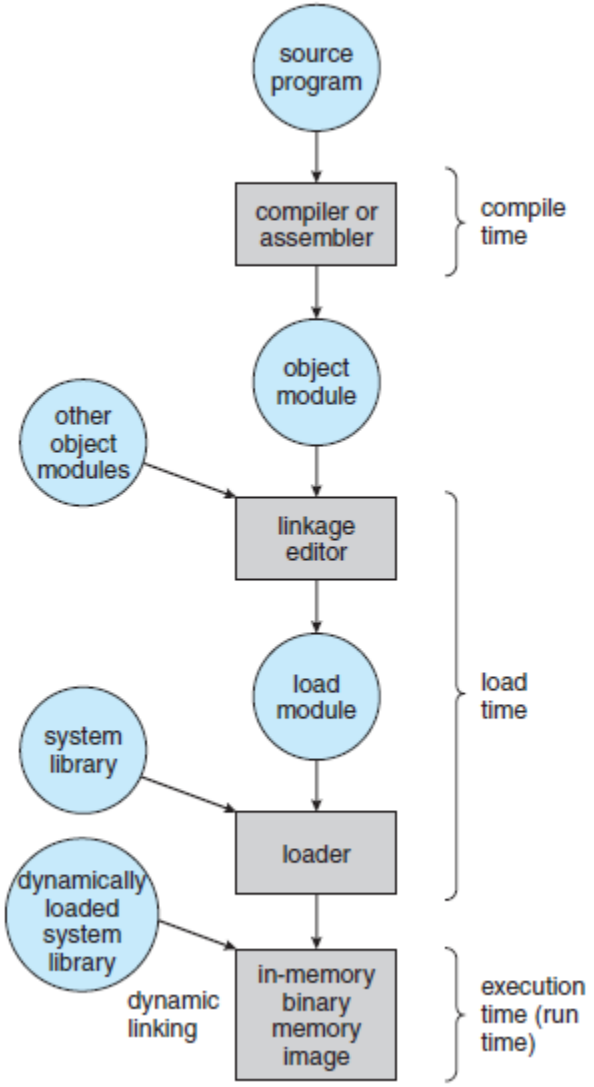
Dobija se memorijska lokacija kojoj je potrebno pristupiti



Provera zaštite – proces sme da referencira samo dozvoljene lokacije

Kreiranje i izvršavanje programa

- Program je binarni fajl na disku
- Da bi se izvršio, ubacuje se u glavnu memoriju
- U toku izvršavanja referencira stvarne adrese u memoriji
- Izvorni kod ne koristi direktno adrese, nego promenljive
- Kada se definišu stvarne adrese?



Definisanje adresa u programu

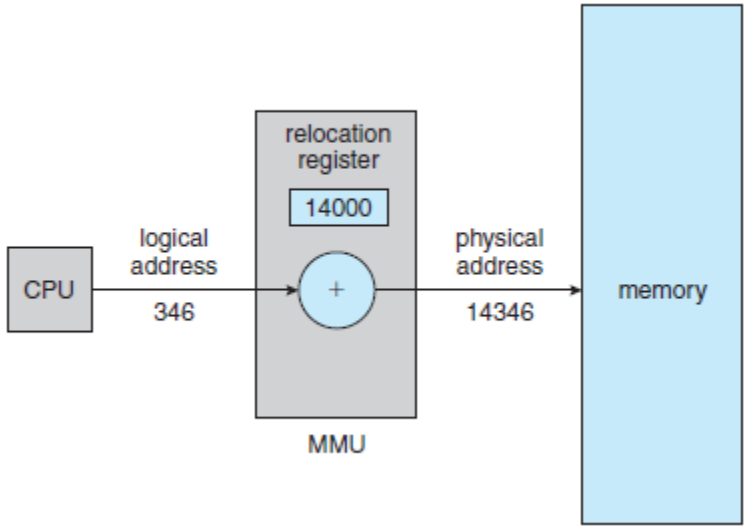
- U trenutku kompajliranja
  - Mora se u trenutku kompajliranja znati gde će program biti smešten u memoriji
  - U samom programu se upišu stvarne memorijske adrese kojima će se pristupati
  - Ako treba podržati multiprogramiranje, nije moguće u trenutku kompajliranja znati lokaciju programa
  - Moglo je u starijim OS (npr. MS-DOS)
- U trenutku učitavanja
  - Stvarne adrese se upisuju tek u trenutku učitavanja programa u memoriju
  - Nije upotrebljivo ako u toku izvršavanja program može da bude pomeran na druge lokacije u memoriji

Definisanje adresa u programu

- U trenutku izvršavanja
  - Ovo je jedina varijanta koja podržava relokaciju
    - Ako nije unapred definisana lokacija na koju će program biti učitan i ako u toku izvršavanja programa ta lokacija može da se menja, tek u trenutku izvršavanja svake instrukcije se može odrediti stvarna adresa kojoj da se pristupi
  - Potrebno je da hardver procesora podrži utvrđivanje adrese u trenutku izvršavanja
  - Savremeni OS koriste ovu metodu

Logičke i fizičke adrese

- Adresa upisana u programu je logička (relativna) adresa
- Stvarna adresa kojoj se pristupa je fizička (apsolutna) adresa
- Logičke adrese su definisane relativno u odnosu na blok u koji je proces trenutno smešten
- U toku izvršavanja ove relativne adrese se prevode u fizičke adrese u memoriji
- Hardver procesora (jedinica za upravljanje memorijom) prevodi relativne u fizičke adrese



Načini upravljanja memorijom

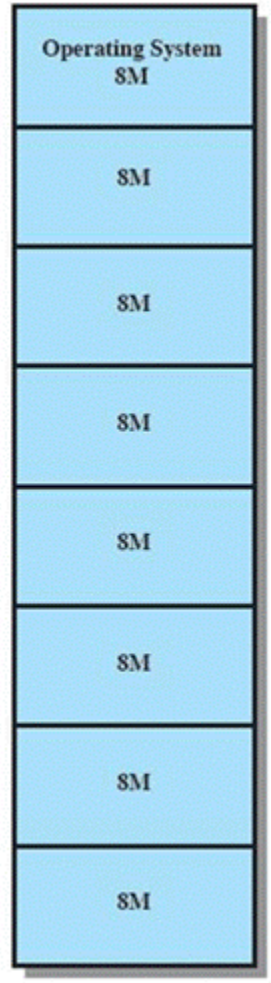
- Alociranje susednih memorijskih lokacija
  - Deljenje memorije na particije
  - Partnerski sistem
- Straničenje
- Segmentacija

Deljenje memorije na particije

- Klasičan način upravljanja memorijom
  - Korišćen pre pojave virtuelne memorije
  - Kompletan proces smešta u uzastopne memorijske lokacije
  - Danas se slabo koristi
- Uvod za razumevanje koncepta virtuelne memorije

Fiksno deljenje na particije – particije jednake veličine

- Podeliti glavnu memoriju u delove (particije)
  - blokovi fiksne veličine
- Svaki proces čija veličina je manja ili jednaka veličini particije može da se smesti u bilo koju raspoloživu particiju
- OS može da zameni proces u particiji
  - Ako treba novi proces da se ubaci u glavnu memoriju, a sve particije su zauzete

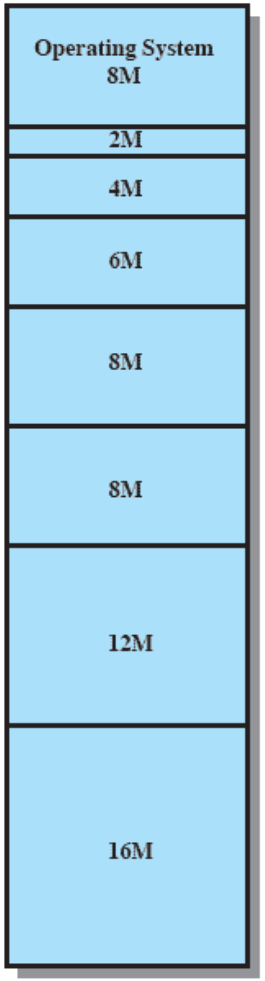


Problemi fiksnog deljenja na particije sa jednakim veličinama particija

- Program može da bude prevelik za particiju
- Neefikasno korišćenje glavne memorije
  - Bez obzira na veličinu programa, zauzima se cela particija
  - To je **interna fragmentacija**

Fiksno deljenje na particije – particije nejednake veličine

- Umanjuju pomenute probleme
  - Ne rešavaju ih potpuno
- Na slici
  - Programi manji od 16M mogu da se smeste u memoriju
  - Manji programi mogu da se smeste u manje particije, čime se smanjuje interna fragmentacija

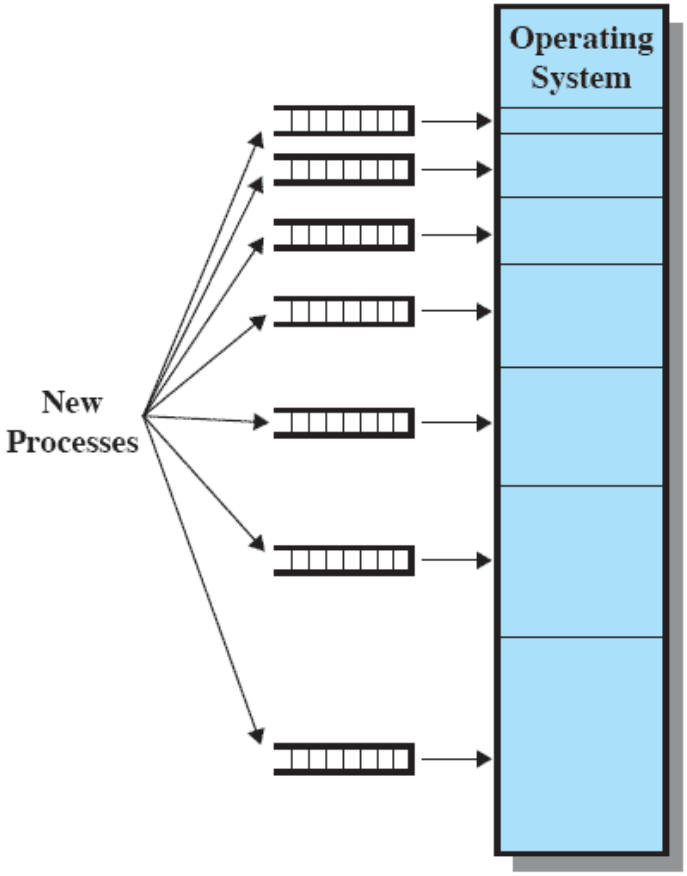


Fiksno deljenje na particije – algoritam smeštanja

- Particije jednake veličine
  - nije važno koja particija će biti zauzeta, jer su sve jednake veličine
  - zauzima se prva raspoloživa particija

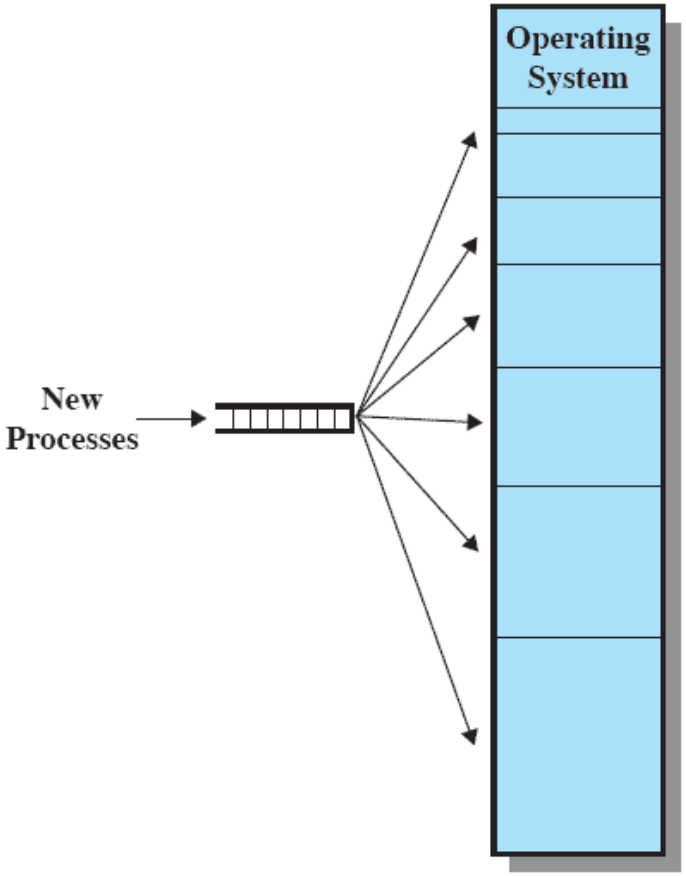
Fiksno deljenje na particije – algoritam smeštanja

- Particije nejednake veličine
  - Procesu se dodeljuje najmanja particija u koju može da stane
    - Za svaku particiju, red čekanja
    - Neefikasno, jer proces može da čeka i kad ima slobodna particija u koju može da se smesti



Fiksno deljenje na particije – algoritam smeštanja

- Particije nejednake veličine
  - Efikasnija varijanta
    - Jedan red čekanja za sve particije
    - Bira se najmanja raspoloživa particija



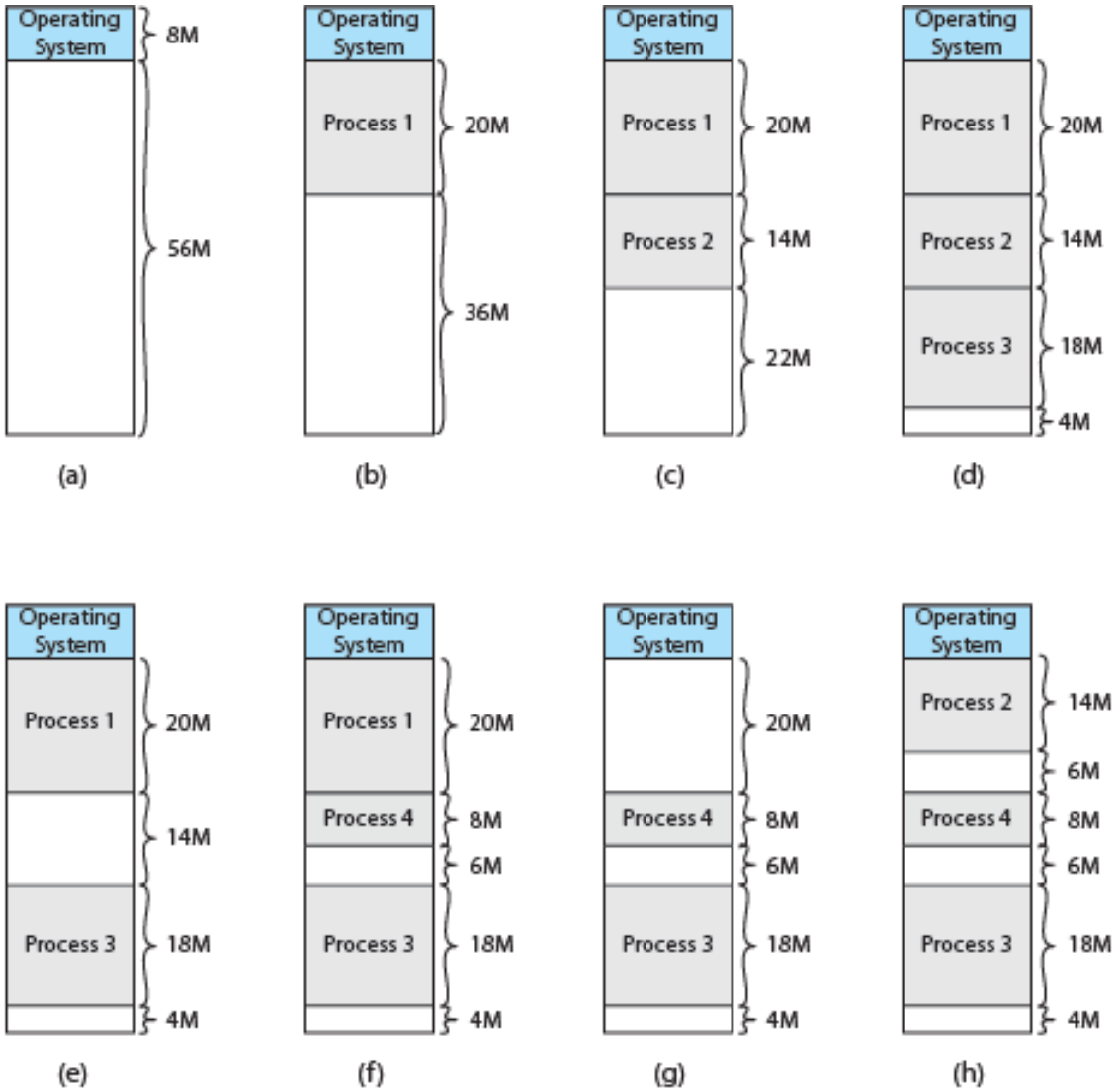
Nedostaci fiksnog deljenja na particije

- Ograničen broj aktivnih procesa
  - Predefinisanim brojem particija ograničeno je koliko procesa može biti u radnoj memoriji
- Veliki broj malih procesa neće efikasno koristiti memoriju
  - Značajni delovi particija će biti neiskorišćeni

Dinamičko deljenje na particije

- Promenljiv broj particija
- Particije promenljive veličine
- Procesu se dodeljuje tačno onoliko memorije koliko zahteva

Dinamičko deljenje na particije



Dinamičko deljenje na particije

- Eksterna fragmentacija
  - mnogo malih rupa u memoriji
  - dovoljno slobodne memorije, ali nije u povezanim lokacijama
  - proces ne može da se smesti iako ukupno ima dovoljno memorije za smeštanje
  - Može se rešiti sažimanjem
    - s vremena na vreme se pomeraju procesi tako da budu u susednim lokacijama
    - sva slobodna memorija bude na kraju u jednom bloku
    - traje dugo i troši procesorsko vreme

Dinamičko deljenje na particije – algoritam smeštanja

- OS mora da odluči u koji blok da smesti proces
- Najbolji odgovarajući

Izvor: www.youtube.com

Dinamičko deljenje na particije – algoritam smeštanja

- Najbolji odgovarajući
  - Bira se slobodan blok koji je po veličini najbliži zahtevu
  - Najlošije performanse
  - Najmanji mogući deo bloka ostaje neiskorišćen
  - Sažimanje mora češće nego kod ostalih algoritama, jer ostaje puno malih blokova

Dinamičko deljenje na particije – algoritam smeštanja

- Prvi odgovarajući

Izvor: www.youtube.com

Dinamičko deljenje na particije – algoritam smeštanja

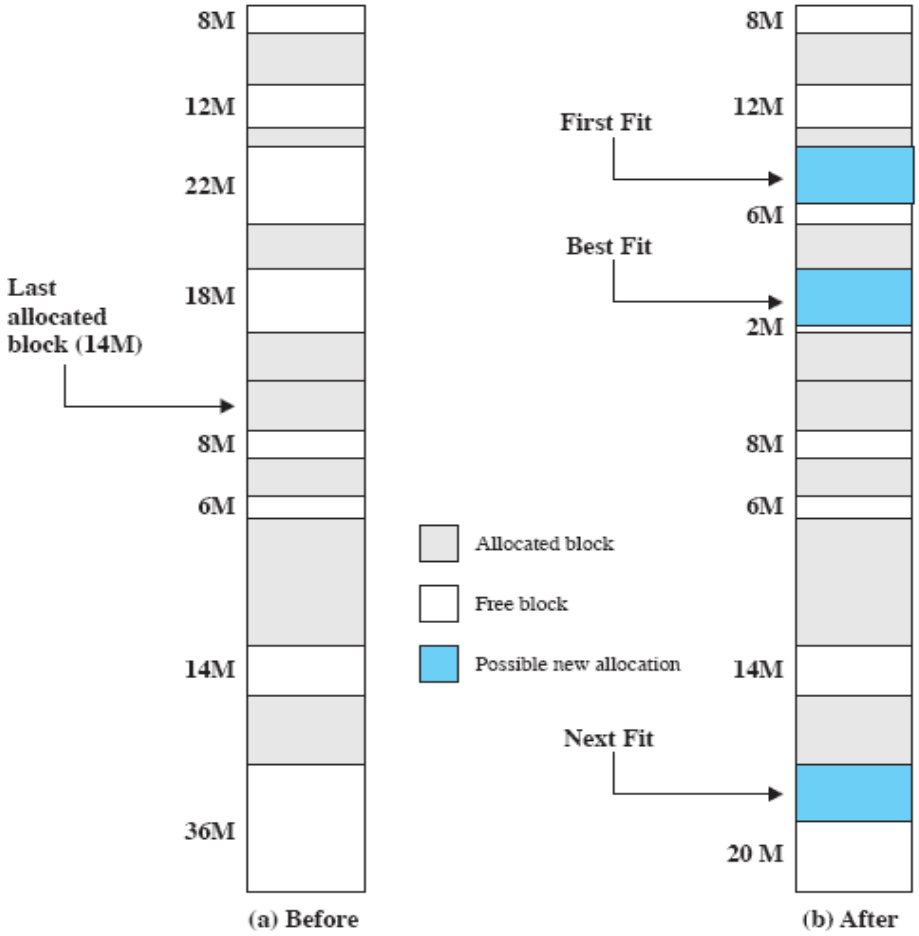
- Prvi odgovarajući
  - Prolazi se kroz memoriju od početka
  - Pronalazi se prvi slobodan blok dovoljno velik da se proces smesti
  - Najjednostavniji i najbrži algoritam
  - Može da rezultira u mnogo malih slobodnih particija u početnom delu memorije, koje se uvek moraju iznova pretraživati

Dinamičko deljenje na particije – algoritam smeštanja

- Sledeći odgovarajući
  - Prolazi se kroz memoriju počevši od lokacije na kojoj je izvršeno poslednje ubacivanje
  - Često procese smešta na kraj memorije gde je najveći slobodan blok
  - Ovaj blok će biti izdeljen na male fragmente
  - Potrebno je sažimanje da bi se opet dobio veliki slobodan blok na kraju memorije

Dinamičko deljenje na particije – algoritam smeštanja

- Ubacuje se blok od 16M



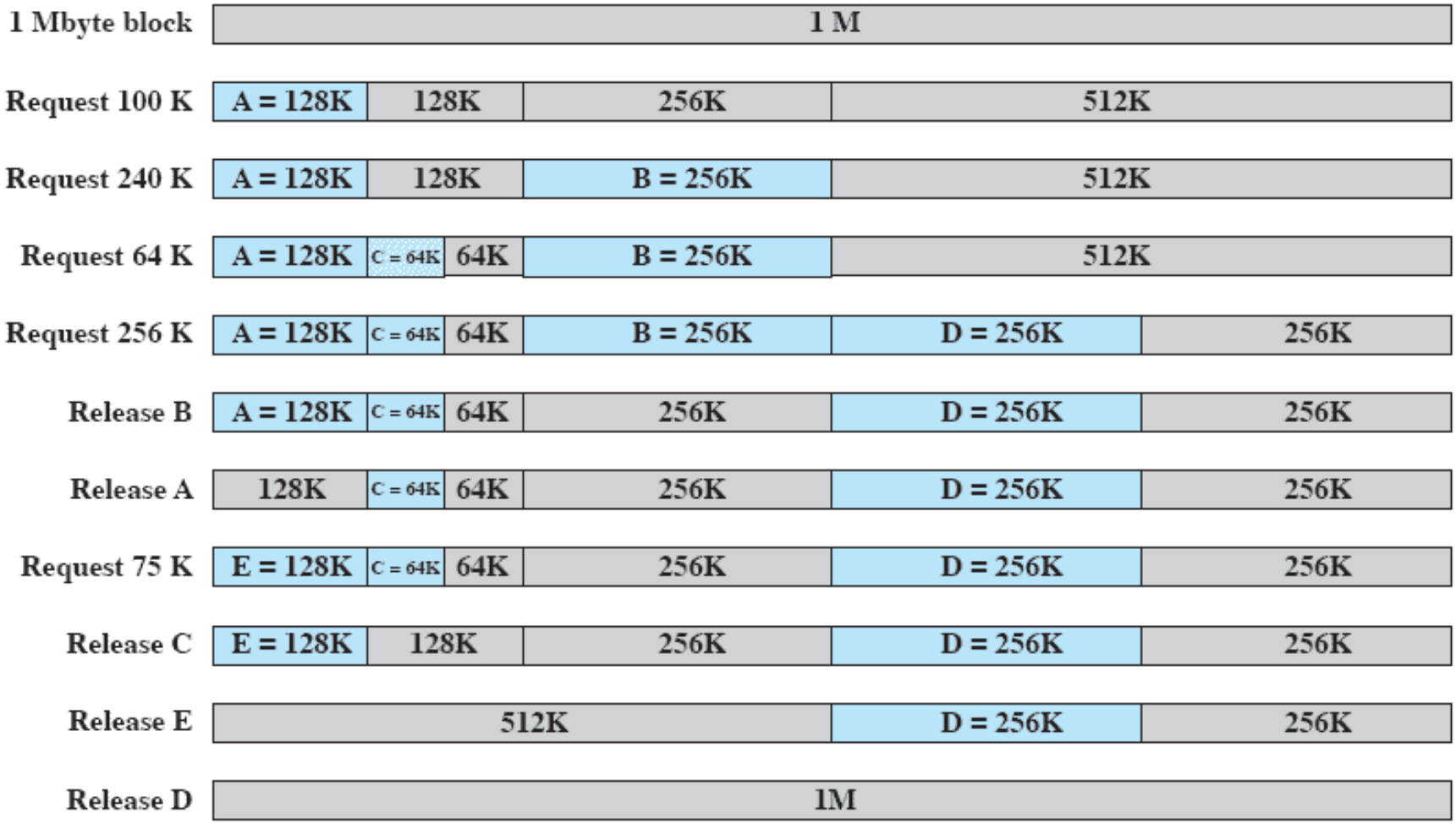
Deljenje na particije

- Problem interne i eksterne fragmentacije
- Npr. kod dinamičkog deljenja na particije i korišćenja algoritma „prvi odgovarajući“ za N alociranih blokova još N/2 blokova će biti izgubljeno zbog eksterne fragmentacije
- Ne koristi se u savremenim OS

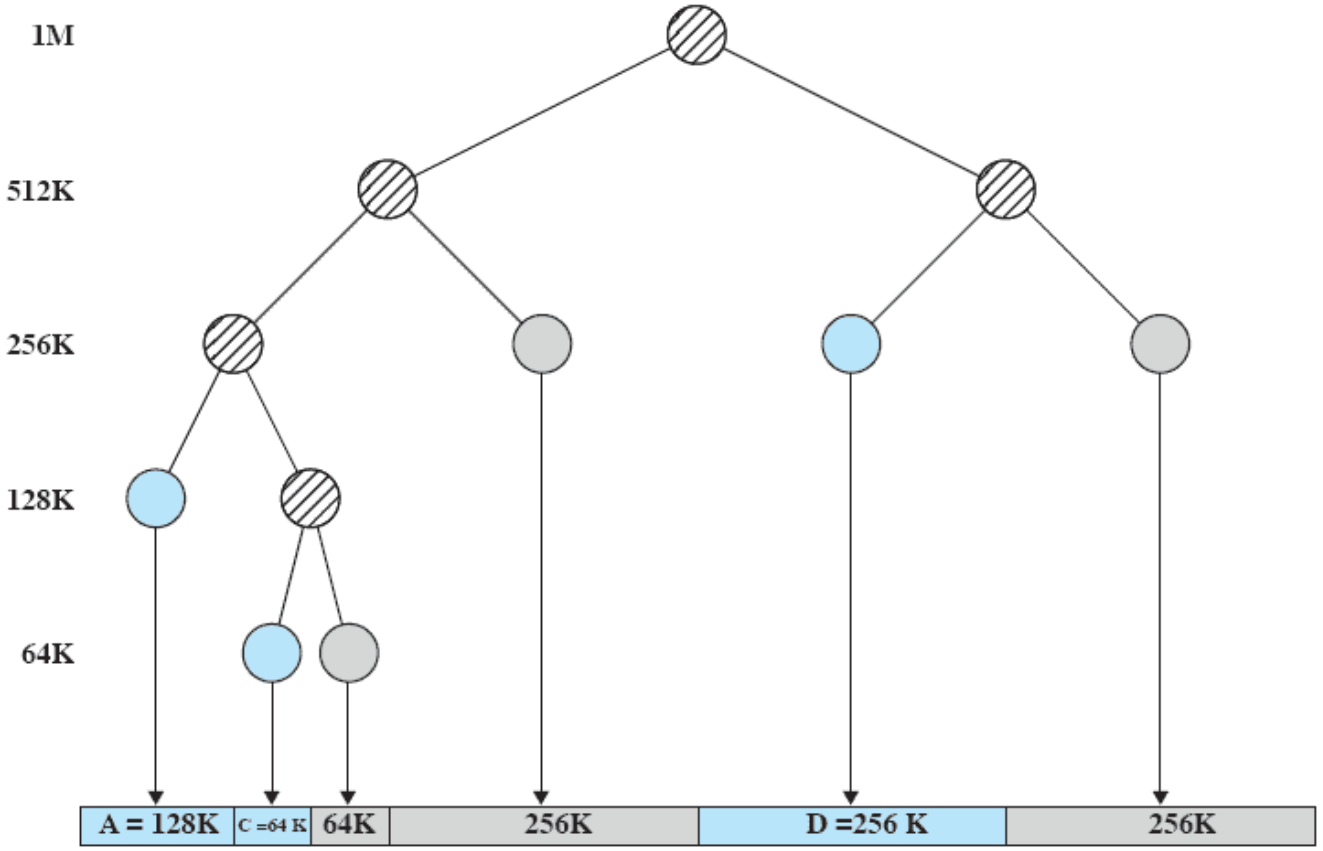
Partnerski sistem

- Celokupna raspoloživa memorija se tretira kao jedan blok veličine 2^U
- Ako se zahteva blok veličine *s*, $2^{U-1} < s <= 2^U$
  - ceo blok se alocira
- U suprotnom, blok se deli na dva jednaka bloka (partnera)
- Postupak se rekurzivno ponavlja dok god se ne stvori najmanji blok koji je veći ili jednak *s*

Primer rada partnerskog sistema



Primer rada partnerskog sistema

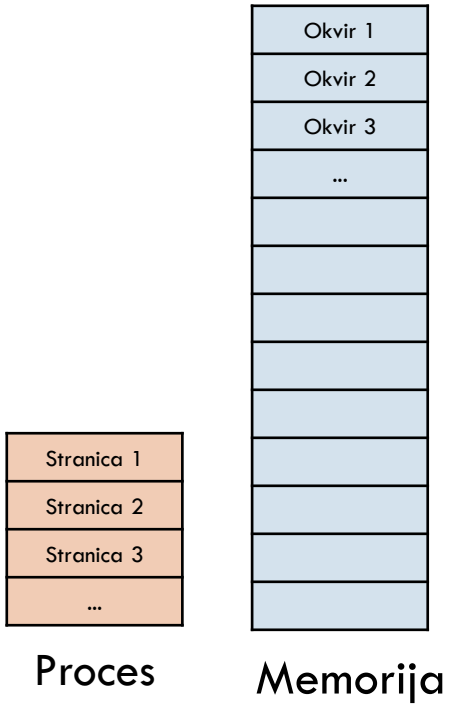


Partnerski sistem

- Prednosti
  - Zauzima se blok koji najbolje odgovara veličini procesa
  - Jednostavno se sažimaju susedni slobodni blokovi
- Nedostaci
  - Interna fragmentacija
- U savremenim OS se kao osnovni način upravljanja memorijom koriste
  - Straničenje i
  - Segmentacija
- Modifikovana varijanta partnerskog sistema se koristi jednim delom u Linuxu u kombinaciji sa straničenjem

Straničenje

- Memorija se deli na male delove jednake veličine
- Proces se deli na male delove jednake veličine
- Delovi memorije se zovu **okviri**
- Delovi procesa se zovu **stranice**
- Veličina okvira i stranice je jednaka
- Stranice procesa se mogu dodeljivati okvirima, **koji ne moraju biti susedni**



Straničenje

- Eksterne fragmentacije nema
- Interna fragmentacija samo u poslednjoj stranici procesa

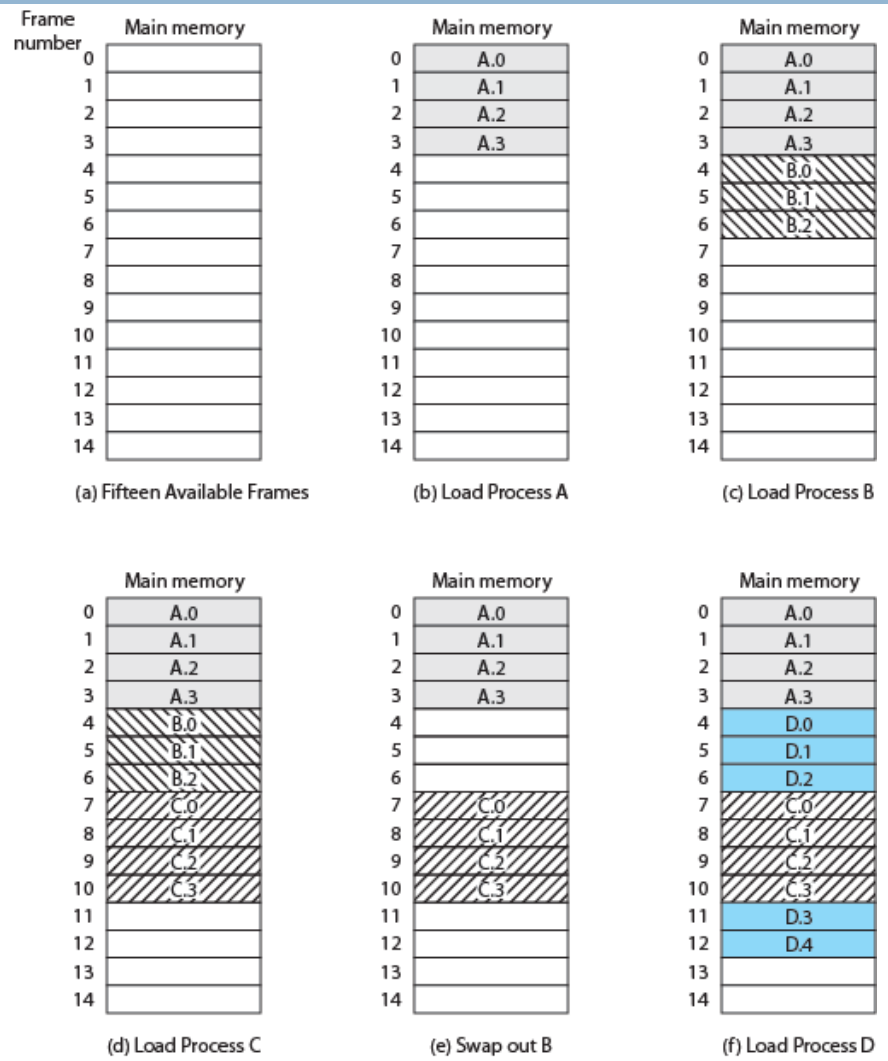
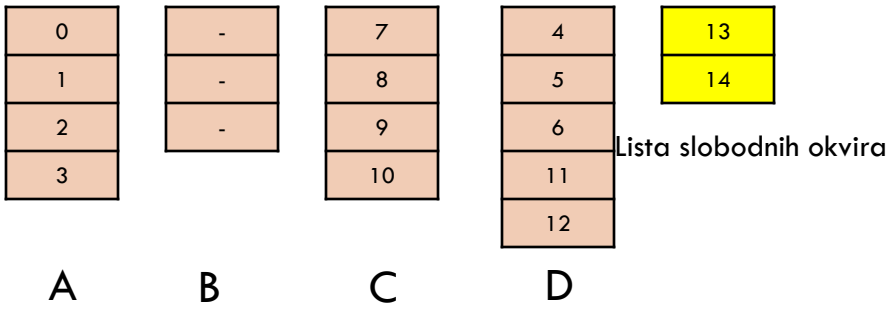


Tabela stranica

- OS za svaki proces održava tabelu stranica
- Za svaku stranicu tabela sadrži broj okvira u koji je smeštena
- Procesor mora da zna da pristupi tabeli stranica procesa
- Na osnovu broja stranice i relativnog pomeraja u okviru stranice, formira se apsolutna adresa

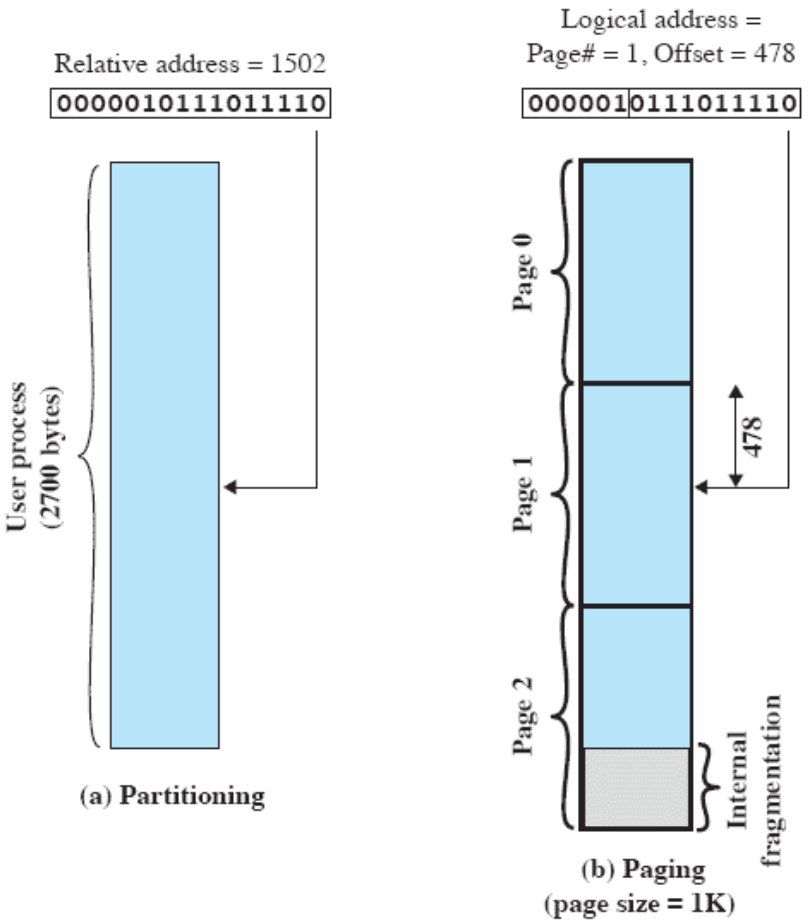
Tabela stranica



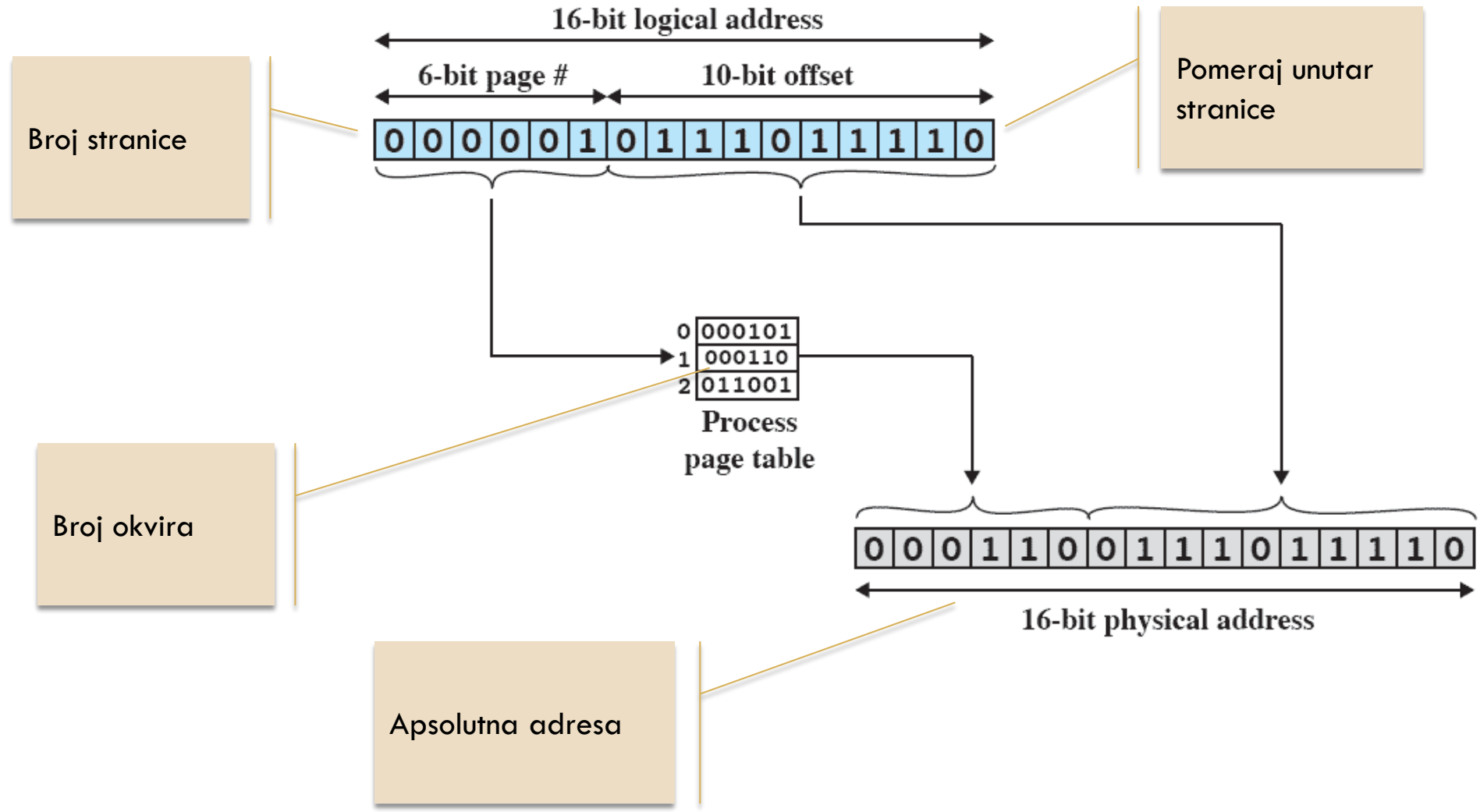
Tabele stranica

Logičke adrese

- Relativna adresa
  - pomeraj u odnosu na početak
- Logička adresa
  - broj stranice i pomeraj unutar nje
- Ako se odredi da je veličina stranice stepen od 2
  - tada su relativna i logička adresa iste

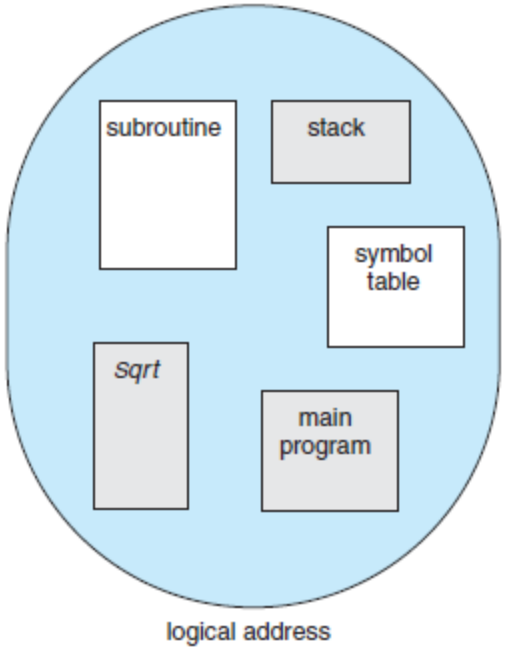


Straničenje – prevođenje logičke adrese u fizičku



Segmentacija

- Programer „ne vidi“ program kao niz adresiranja linearno organizovanih memorijskih lokacija
- Programer vidi program kao skup **segmenata**
  - segmenti su različite dužine
  - postoji ograničenje za maksimalnu veličinu

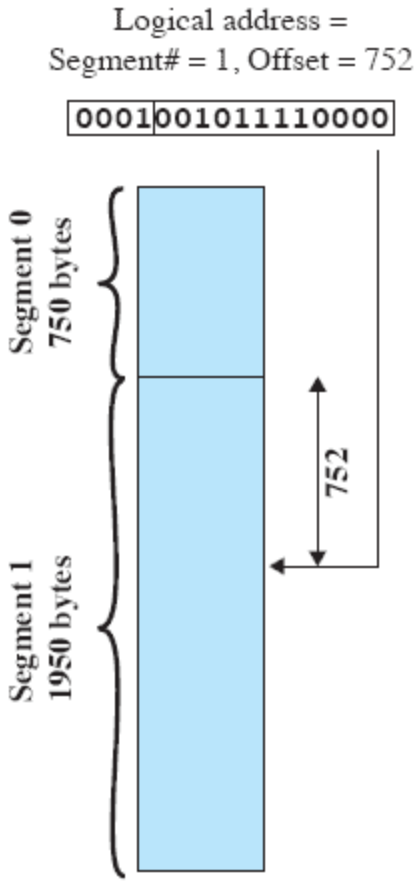


Segmentacija

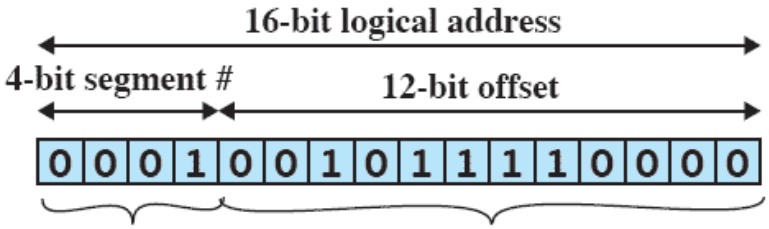
- Segmentacija je način upravljanja memorijom koji se bazira na posmatranju memorije iz ugla programera
- Logički adresni prostor se deli u segmente promenljive veličine
- Kompajler generiše odvojene segmente za različite delove koda
- Slično dinamičkom deljenju na particije
- Nema interne fragmentacije

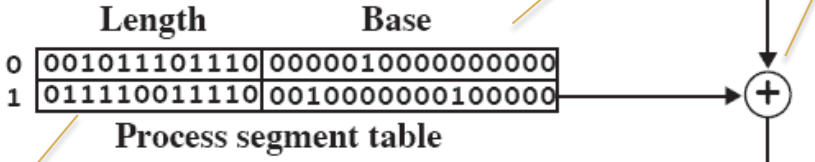
Segmentacija

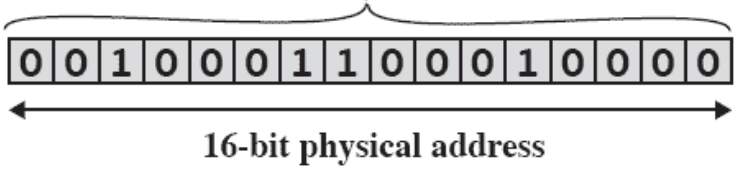
- Adresa se sastoji iz dva dela
  - broj segmenta
  - pomeraj unutar segmenta



Segmentacija – prevođenje logičke adrese u fizičku







Pomeraj unutar segmenta

Početak segmenta

Na početnu adresu segmenta dodaje sa adresa pomeraja

Broj segmenta

Dužina segmenta zbog zaštite pristupa

Tabela segmenata za svaki proces

Apsolutna adresa