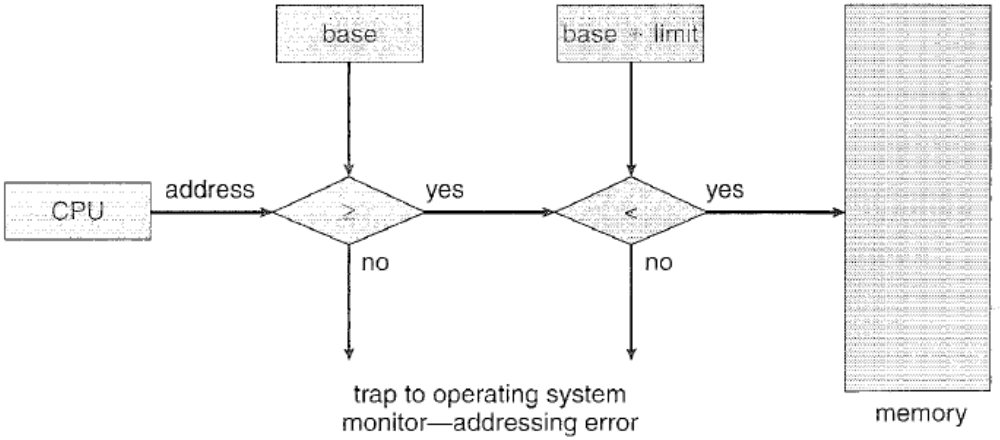
Operativni sistemi
- Upravljanje memorijom -

Veljko Stanković

Uvod

- Da bi smo povecali efikasnost iskoriscenja sistem, potrebno je da drzimo vise procesa istoveremeno u glavnoj memoriji.
- CPU moze direktno da pristupi samo registrima i glavnoj memoriji.
- Adresni prostor procesa mora biti jasno razgranicen
  - Uvode se bazni i granicni registar koji cuvaju najmanju najvecu adresu kojoj neki proces moze da pristupi.
  - Samo OS moze da menja vrednost, ucitava i odredjuje adresu na osnovu ovih registara.



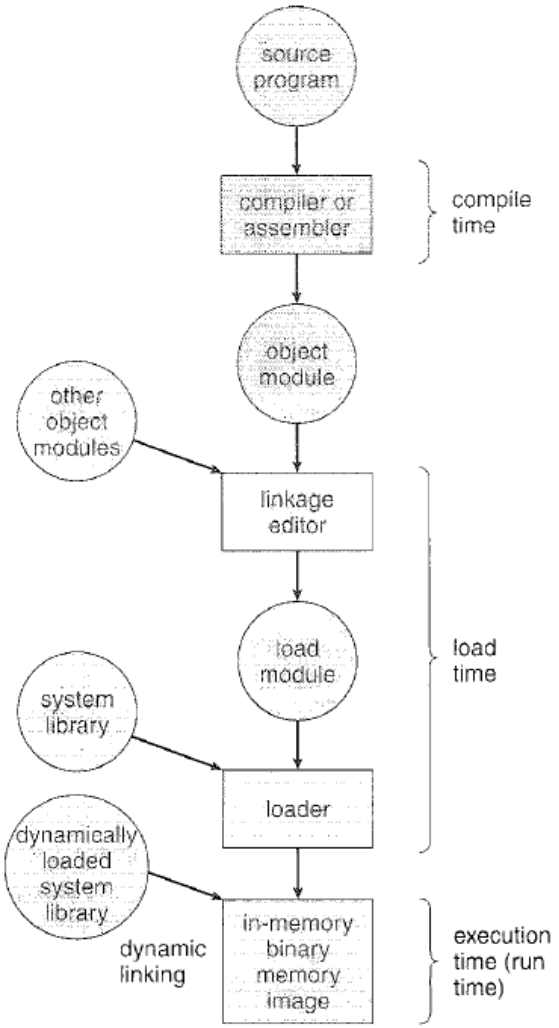
Povezivanje adresa

1010011
1110100
1100001
1010110

- Povezivanje instrukcija/podataka i adresa u memoriji se moze vrsiti:
  - Tokom kompilacije (compile time)
    - ☑ Ako tokom kompilacije znamo gde ce proces nalaziti u glavnoj memoriji tada tokom kompilacije generisemo apsolutni kod, tj kod sa fizickim adresama
  - Tokom ucitavanja programa u glavnu memoriju (loade time)
    - ☑ Generise se relokatibilni kod, tj fizicke adrese se generisu tek nakon sto se program ucita u glavnu memoriju
  - Tokom izvrsavanja
    - ☑ Ako tokom izvrsavanja menjamo lokaciju procesa u glavnoj memeoriji, tada moramo odloziti proces odredjivanja fizickih adresa do trenutka kada proces pocne sa izvrsavanjem.
    - ☑ Potreban poseban hardver za njegovu implementaciju.

Povezivanje adresa

1010011
1110100
1100001
1010110



Logicke i fizicke adrese

1010011
1110100
1100001
1010110

- Logicke adrese generise CPU.
- Fizicke adrese se odnose na konkretne lokacije reci bajtova u glavnoj memoriji.
- Ako se povezivanje adresa vrsi tokom komilacije ili ucitavanja, tada su logicke i fizicke adrese iste.
- Ako se povezivanje adresa vrsi tokom izvrsenja programa tada se logicke i fizicke adrese razlikuju.
  - Virtuelne adrese
  - Skup svih logickih adresa formira logicki adresni prostor
  - Skup svih fizickih adresa formira fizicki adresni prostor
- MMU (Memory management unit)
  - Vrsi prevodjenje logickih (virtuelnih) adresa u fizicke

Dinamicko ucitavanje i povezivanje

- Da ne bismo morali da ucitavamo (punimo) cele procese u glavnu memoriju, ucitavamo samo galvni program a ostale rutine u boliku relokatibilnog koda sotaju na HDD i ucitavaju se po potrebi.
- Kod statickog povezivanja (staticke biblioteke) dodatne rutine se ucitavaju zajedno sa glavnim programom u GM.
- Kod dinamickog povezivanja, dinamicke biblioteke se povezuju sa programom tek kada budu pozvane od strane glavnog programa.
  - Svakom programu se dodaje **stub**, mali deokoda koji ukazuje kako mozemo da odredimo lokaciju rutina koje su rezidentne u GM.
  - Ako rutina nije rezidentna u GM, ona se ucitava.
  - Stub se zamenjuje adresom rutine i rutina se izvrsava.
- Swapping
  - Proces se moze privremeno prekinuti sa izvrsavanjem i smestiti na HDD.

Kontinualna alokacija memorije

1010011
1110100
1100001
1010110

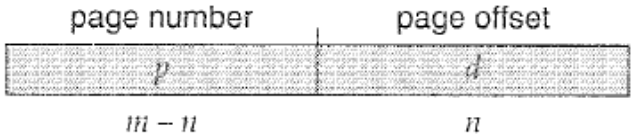
- Cilj je da u GM drzimo vise procesa koji se konkurentno izvrsavaju
- Glavna memorija se deli na particije i svakom procesu se dodeljuje jedna particija.
  - OS prati koji delovi GM su zauseti a koji su slobodni.
  - Kada proce zapocne sa izvrsavanjem, OS trazi deo GM koji je dovoljno veliki da u njega smesti dati proces.
- Algoritmi za odabir dela GM u koji ce se smestiti proces
  - First-fit
    - ☑ Smesti proces u prvi adresni prostor koji je dovoljno veliki.
  - Best-fit
    - ☑ Smesti proces u najmanju particiju koja je dovoljno velika.
  - Worst-fit
    - ☑ Dodeli procesu najvecu particiju.

Kontinualna alokacija memorije

- Fragmentacija memorije
  - Eksterna fragmentacija
    - ☑ Tokom upisivanja i brisanja procesa iz GM javljaju se particije koje su veoma male a koje zahtevaju dosta prostora da bi se sacuvale informacije i njihovoj lokaciji i velicini.
  - Interna fragmentacija
    - ☑ Ako bi nakon dodele neke particiji procesu, preostali deo posmatrane particiji bio isuvise mali, tada se ta particijia ne deli vec se procesu dodeljuje kao celina.
    - ☑ Delovi GM se dodeljuju procesima ali se oni ne koriste.
- Potrebno je izvrsiti kompakciju GM (Compaction)
  - Procesi se smestaju u particije GM koji cine jedan kontinualni adresni prostor.

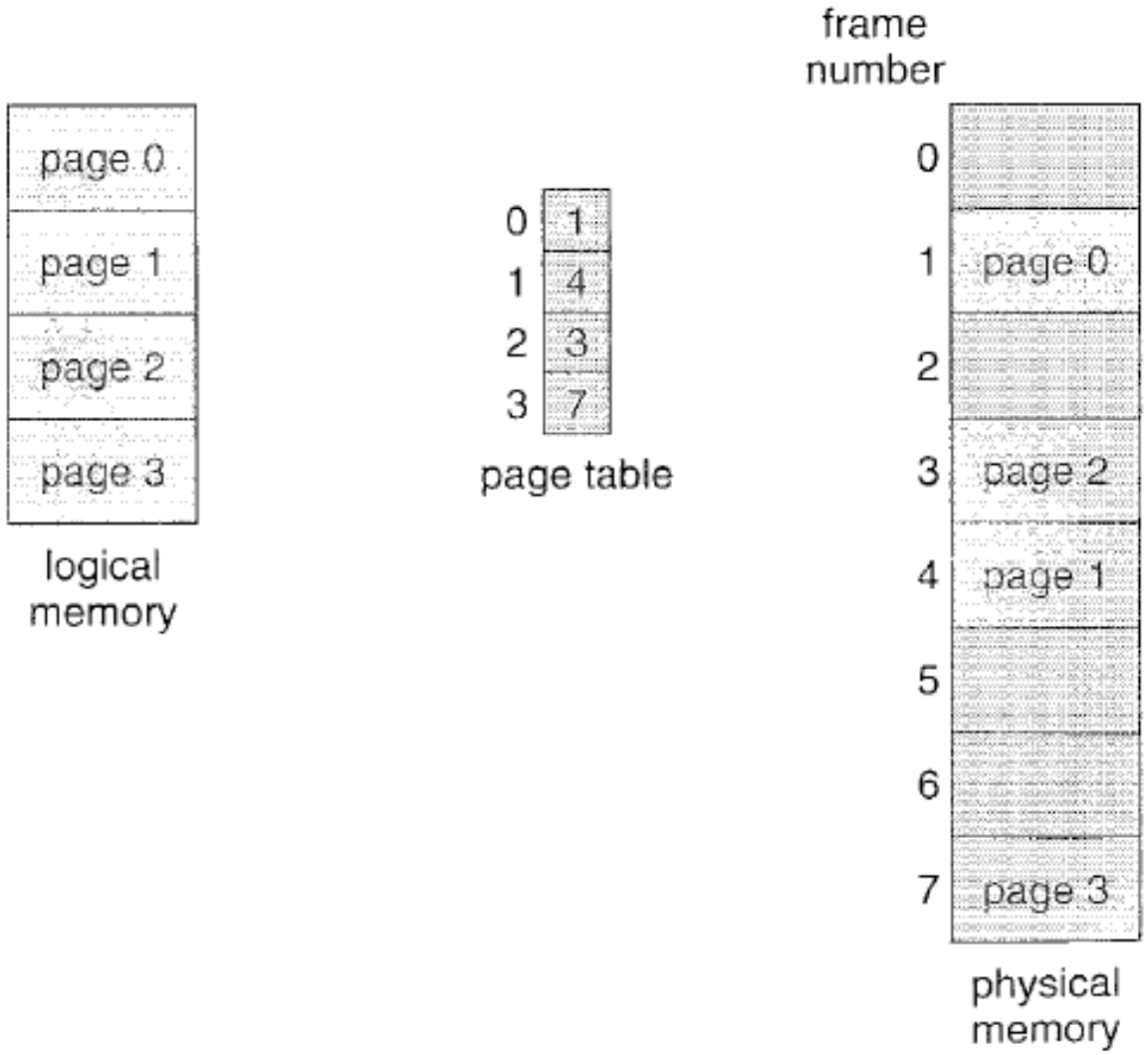
Stranicenje

- Primenom stranicenja adresni prostor procesa ne mora da bude kontinualan.
- Fizicka memorija se deli na blokove fiksne velicini koje nazivamo ramovima.
- Logicki adresni prostor se deli na blokove iste velicine koji se zovu stranice.
- Kada se proces izvrsava, njegove stranice se ucitavaju u bilo koji ram koji je slobodan.
- Logicka adresa se sastoji iz **broja stranice** (p) i **ofset** adrese (d)
  - Broj stranice se koristi kao indeks tabele stranicenja
  - Tabela stranicenja sadrzi baznu adresu svakog rama fizicke memorije.



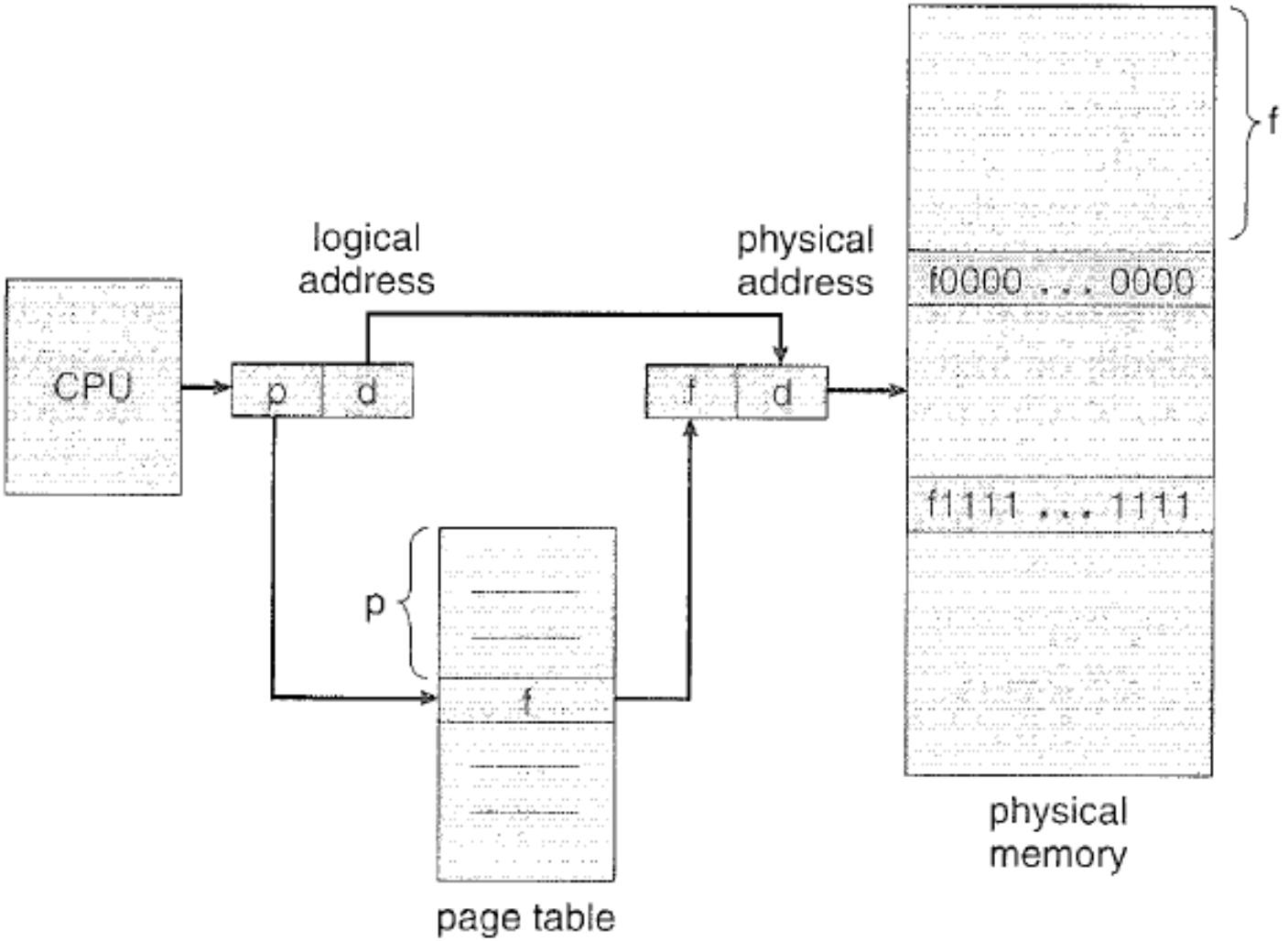
Stranicenje

1010011
1110100
1100001
1010110



Stranicenje

1010011
1110100
1100001
1010110



Stranicenje

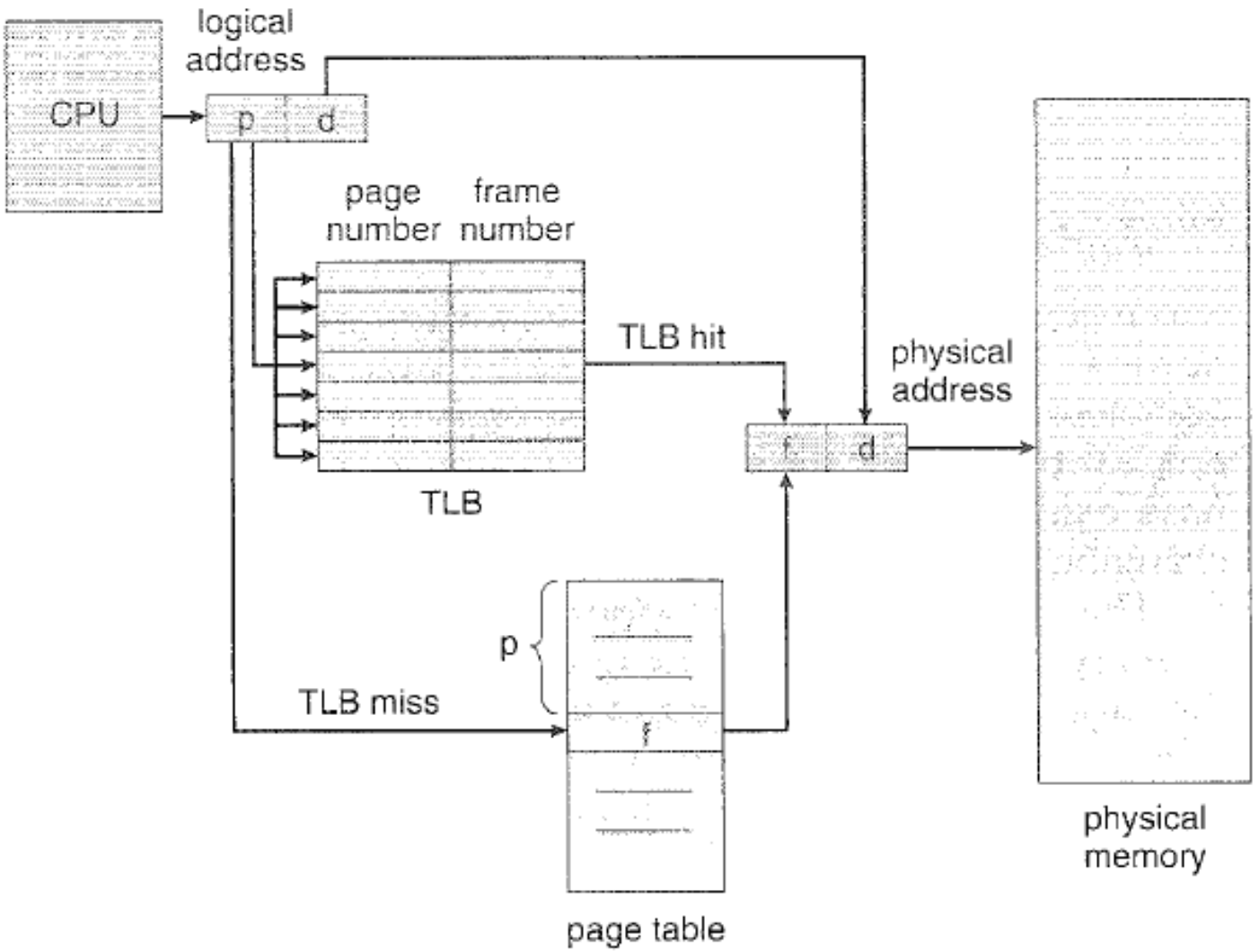
- Kod stranicenja postoji jasna razlika izmedju toga kako korisnik vidi memoriju i stvarne fizicke memorije.
  - Tabela stranicenja svakog procesa sadrzi adrese samo onih ramova koji su dodeljni tom procesu
  - Korisnicki program po definiciji nemoze da pristupi memoriji koja mu nije dodeljena
- Posto OS upravlja fizickom memorijom, mora da vodi racuna o tome koji su ramovi dodeljeni, koji ramovi su slobodni, koliko ukupno ramova ima...
  - Ovi podaci se cuvaju u tabeli ramova (frame table)
- Svaki OS ima svoj nacin cuvanja i azuriranja tabela stranicenja.
  - Uglavnom se svakom procesu dodeljuje po jedna tabela stranicenja.
  - Pokazivac na tabelu stranicenja se smesta zajedno sa ostalim podacima u kontrolni blok procesa (PCB).

Stranicenje

- Moguce je tabelu stranicenja cuvati na HDD, a bazni registar tabele stranicenja pokazuje na tu tabelu (ptbr – page table base register)
  - Zahteva dva ciklusa pristupa HDD ⇒ veoma sporo resenje
- Koristi se kes stranicenja
  - TLB (Translation look-aside buffer)
  - Asocijativna veoma brza memorija
  - Prvo se proverava da je odgovarajuci ulaz tabele stranicenja u TLB (TLB pogodak)
  - Ako se trazena stranica ne nalazi u TLB, vrsi se pristup HDD i ucitava odgovarajuci podataka iz tabele stranicenja, a sam ulaz se unosi u TLB

Stranicenje

1010011
1110100
1100001
1010110

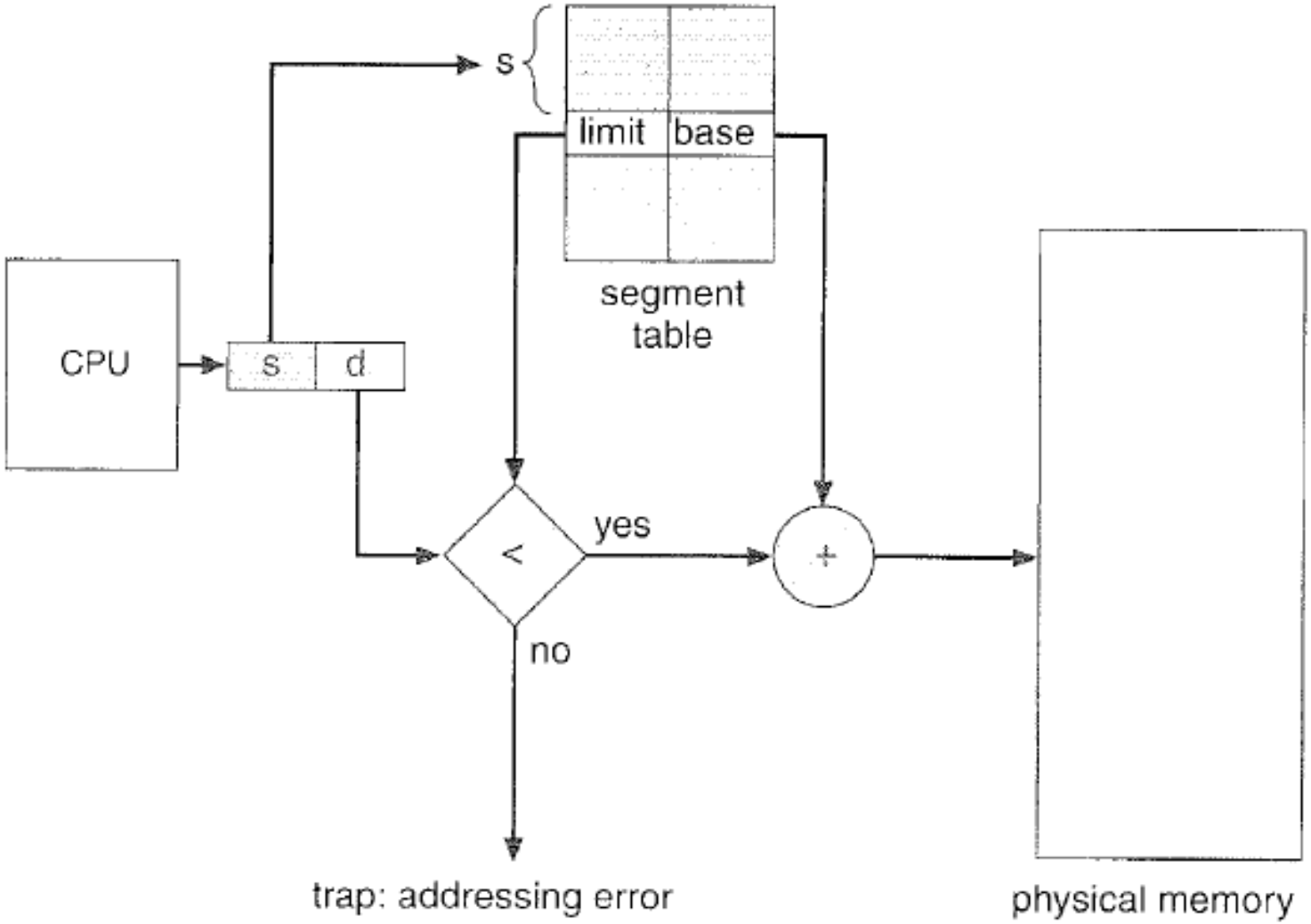


Segmentacija

1010011
1110100
1100001
1010110

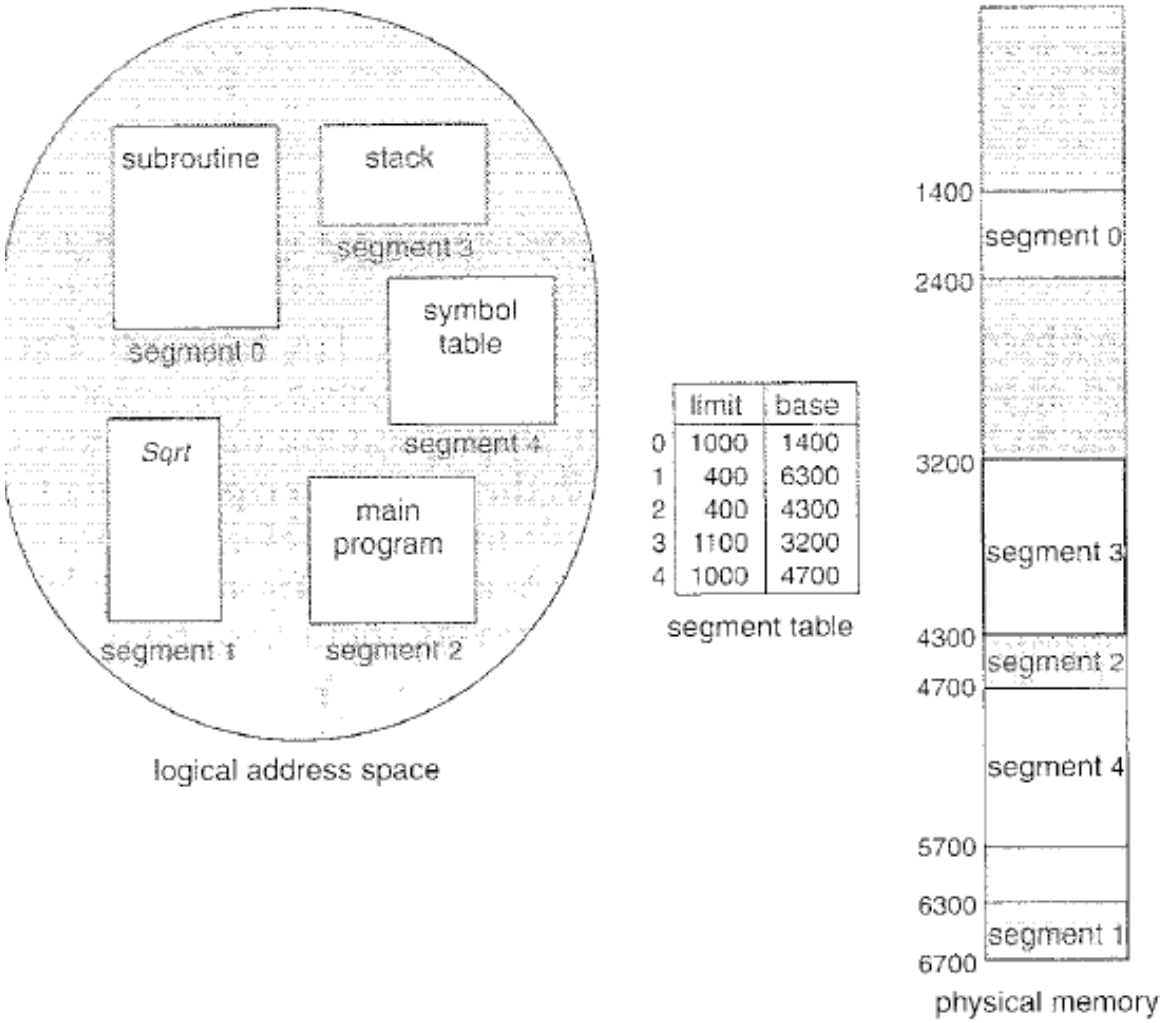
- Memorija se sastoji iz skupa segmenata razlicite velicine.
  - Odgovara korisnickom pogledu na memoriju u koju se smestaju delovi programa razlicite velicine.
- Podaci o segmentima se cuvaju u tabeli semgentacije
  - Bazna adresa segmenta
  - Limit (velicina segmenta)
- Logicka adresa se sastoji iz:
  - Broja segmenta *s*,
  - Ofseta u okviru tog segmenta *d*.
- Kada je vrednost ofseta u dozvoljenim granicama, kombinuje se sa baznom adresom segmenta da bi formirao fizicku adresu.

Segmentacija



Segmentacija

1010011
1110100
1100001
1010110



Domaci zadatak

1010011
1110100
1100001
1010110