Operativni sistemi

- Virtuelna memorija -

Prof. dr Dragan Stojanović

Katedra za računarstvo
Univerzitet u Nišu, Elektronski fakultet

Literatura

- *Operating Systems: Internals and Design Principles*, edition, W. Stallings, Pearson Education Inc., 7th – 2012, (5th -2005, 6th - 2008, 8th – 2014 , 9th – 2017)
  - http://williamstallings.com/OperatingSystems/
  - http://williamstallings.com/OperatingSystems/OS9e-Student/

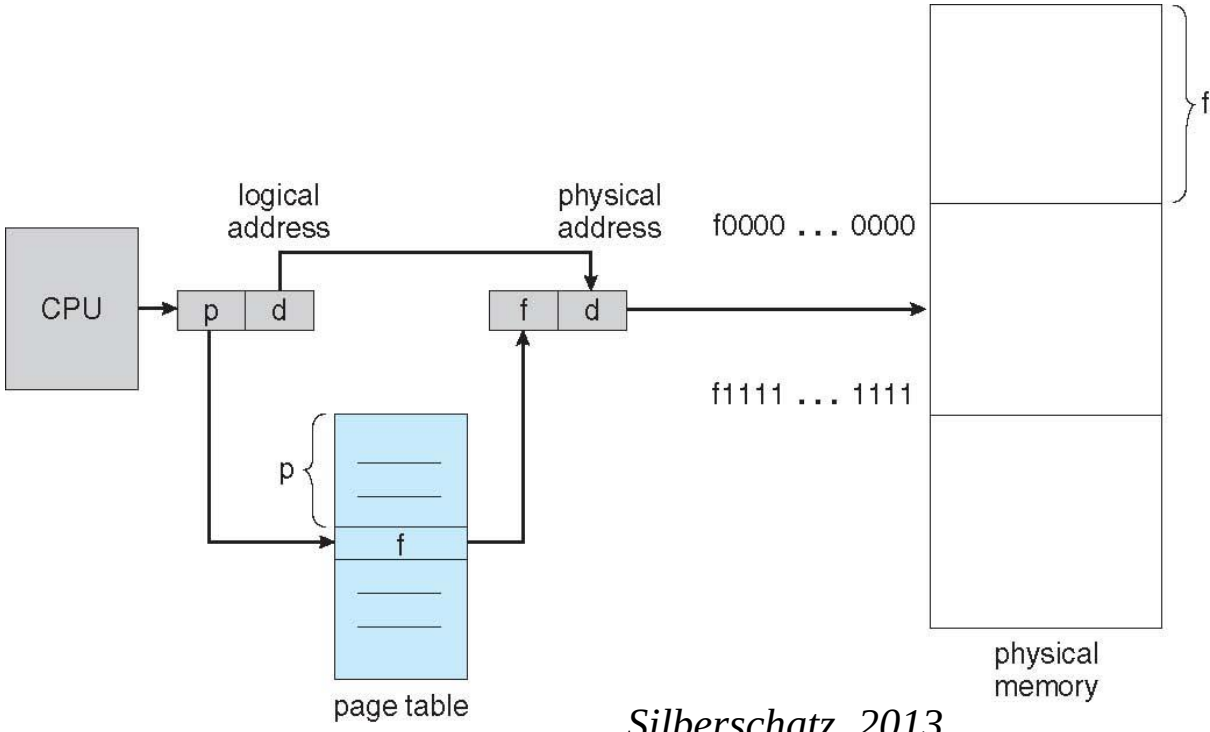
- Poglavlje 8: Virtuelna memorija

Virtuelna memorija

- Nije neophodno da sve stranice ili segmenti procesa budu u glavnoj memoriji u toku izvršavanja
- Logički adresni prostor procesa može biti veći od fizičkog adresnog prostora (raspoložive memorije)
- Samo neophodni delovi procesa su u memoriji (rezidentan skup), a ostatak je na disku
- Kada proces u svom izvršavanju referencira logičku adresu u okviru stranice (segmenta) koja nije u glavnoj memoriji, generiše se prekid koji ukazuje na grešku u pristupu memoriji i tražena stranica se smešta u memoriju
  - Straničenje na zahtev (*Demand paging*)
  - Segmentacija na zahtev (*Demand segmentation*)
- Omogućava deljenje adresnog prostora od strane više procesa

Tabela stranica

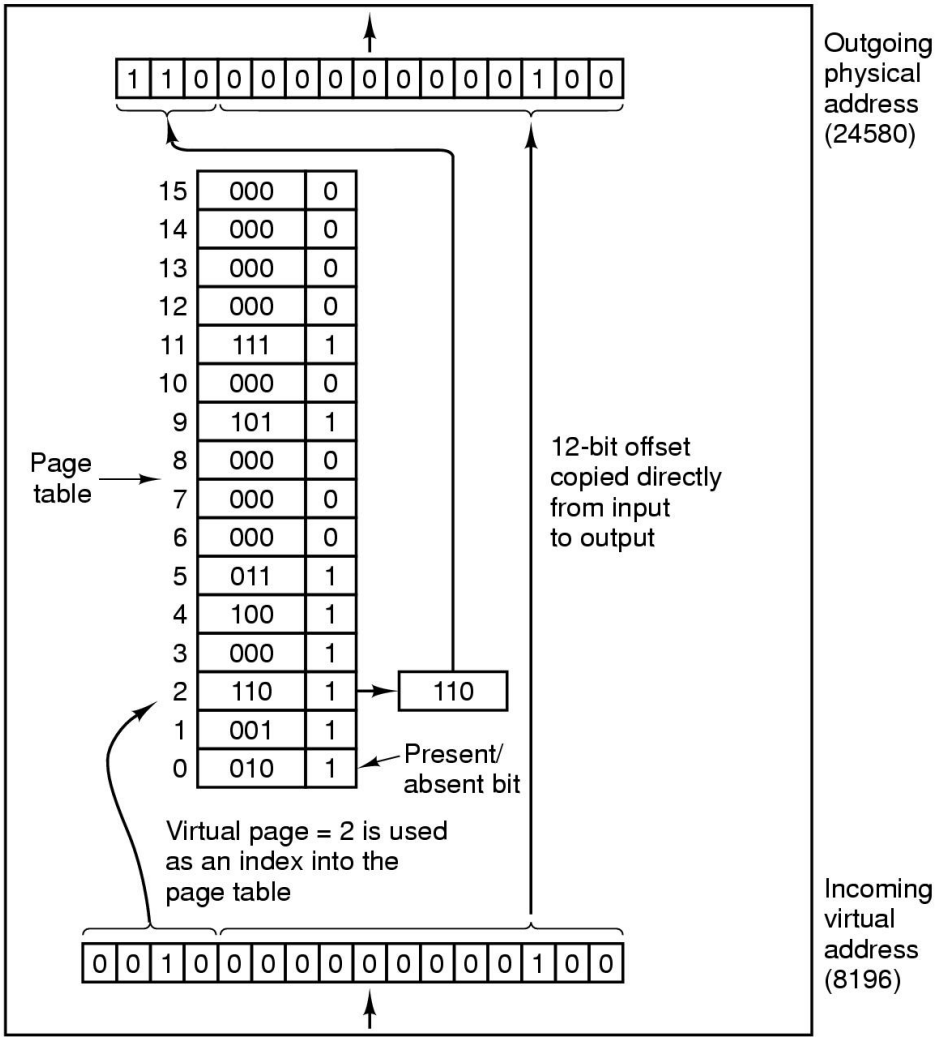
- Virtuelna (logička) adresa sadrži:
  - Broj stranice (bitovi višeg reda)
  - Ofset unutar stranice (bitovi nižeg reda)
- Broj stranice se koristi kao indeks u tabelu stranica u cilju nalaženja odgovarajućeg broja straničnog okvira u određenoj stavci (ulazu, *Page Table Entry*) u tabeli stranica
- Broj straničnog okvira se pridružuje kao viši deo bitovima ofseta i formira fizičku adresu



Silberschatz, 2013

Tabela stranica - primer

- U 16 bitnoj virtuelnoj adresi pri stranicama veličine 4KB,
  - viša 4 bita specificiraju jednu od 16 virtuelnih stranica
  - nižih 12 bitova specificiraju ofset (pomeraj) u okviru stranice (0-4095)
- Bit u tabeli stranica specificira da li je stranica prisutna u memoriji u odgovarajućem straničnom okviru (*Valid-Invalid*, *Present/Absent* bit)

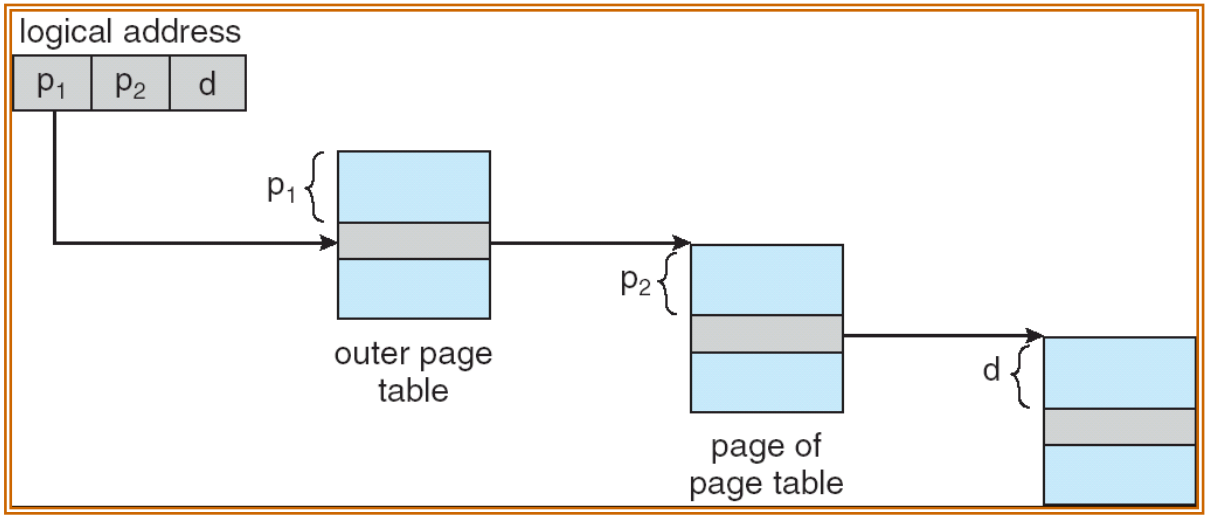


Implementacija tabele stranica

- Tabela stranica je smeštena u glavnoj memoriji
  - Bazni registar tabele stranica *(Page-table base register* - PTBR) sadrži adresu početka tabele stranica
  - Registar veličine tabele stranica *(Page-table length register* - PRLR) sadrži veličinu tabele stranica
- Problemi:
  - Veličina tabele stranica - Za virtuelne adrese od 32 bita i veličinu stranice od 4KB, broj stranica je 1 M, tako da je neophodno isto toliko stavki (ulaza) u tabeli stranica
  - Svaki pristup instrukcijama i podacima procesa zahteva dva pristupa memoriji, jedan za pristup tabeli stranica, a drugi pristup instrukciji/podacima
- Implementacija tabele stranica
  - Kontinualne tabele stranica u
    - Brzim hardverskim registarima
    - Kontinualnom području u memoriji
  - Tabele stranica u više nivoa
  - Invertovane (obrnute) tabele stranica

Tabele stranica u više nivoa

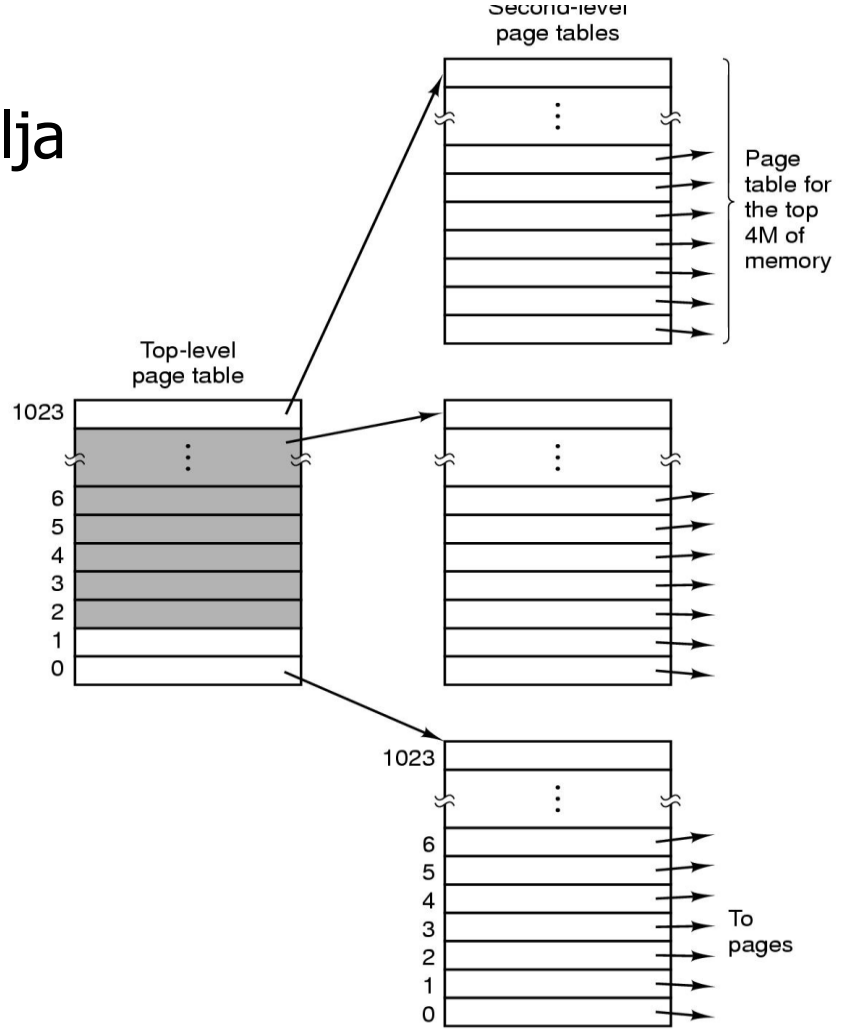
- Tabela stranica se predstavlja u obliku stranica (straniči se)
- Virtuelna adresa se deli na tri polja
  - PT1 – indeks u tabelu stranica prvog nivoa
  - PT2 – indeks u tabelu stranica drugog nivoa
  - Offset – memorijska adresa u okviru stranice
- Nema potrebe čuvati sve tabele stranica u memoriji već samo one potrebne
- Može biti tri, četiri ili više nivoa tabela stranica



Silberschatz, 2013

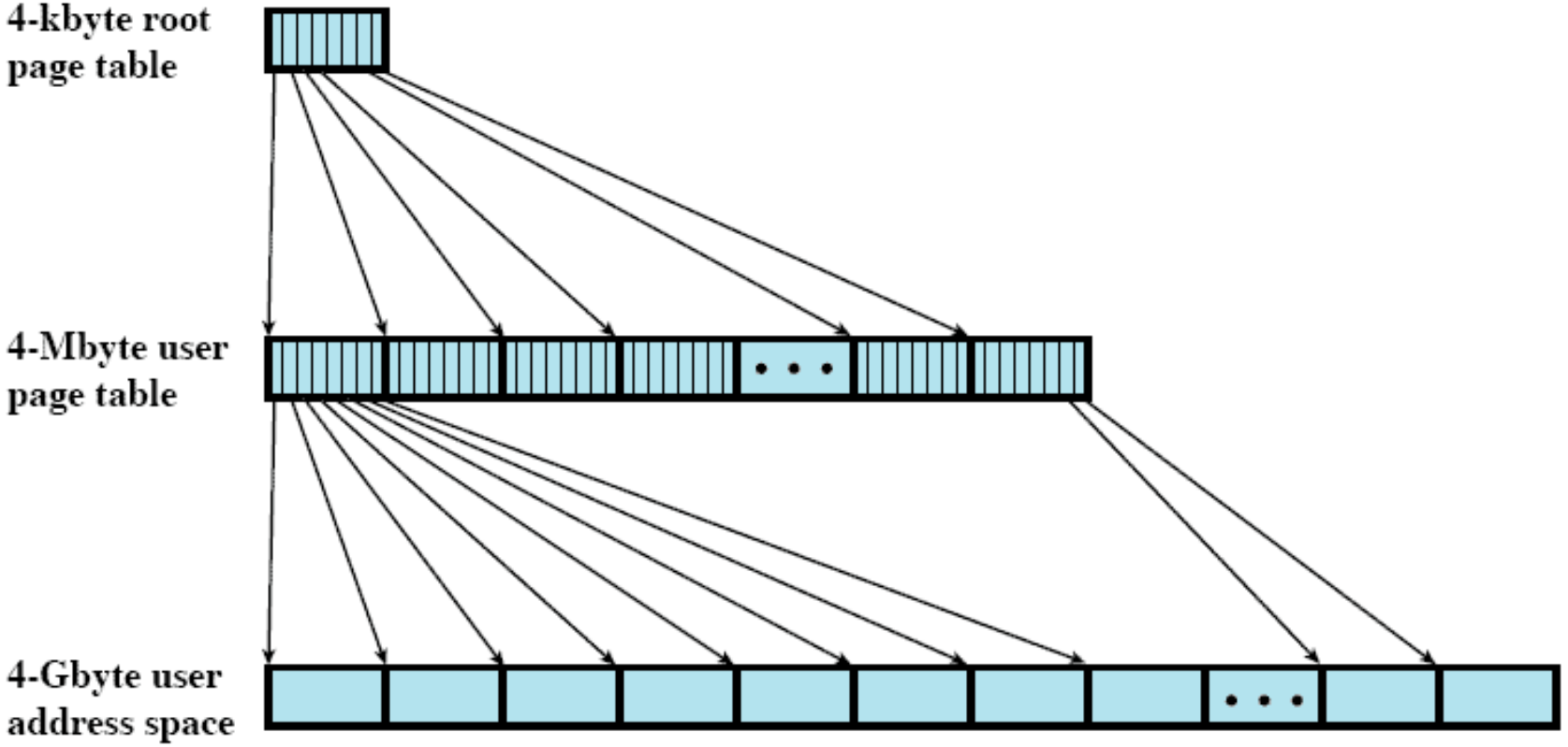
Tabele stranica u dva nivoa

- Stranice veličine 4KB
- 32-bitna adresa se deli na tri polja
  - PT1 – 10 bita
  - PT2 – 10 bita
  - Offset – 12 bitova
- Sve tabele stranica su veličine jedne stranice (4KB), a svaka stavka u tabeli 32 bita
- Za proces adresnog prostora 4MB za kod, 4MB za podatke i 4MB za stek ukupno je potrebno 4 tabele stranica



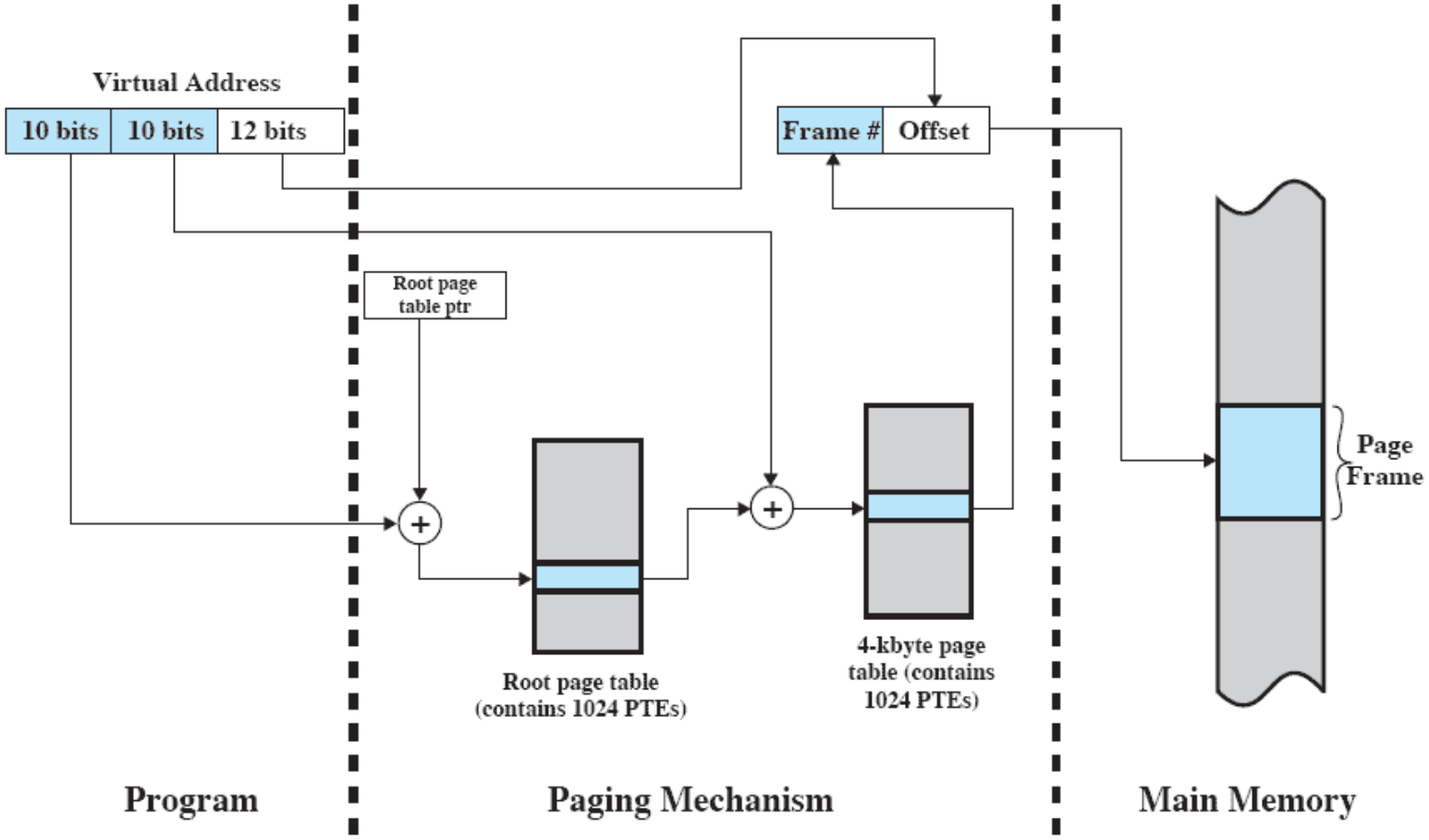
Tabele stranica u dva nivoa (2)

- Hijerarhijska šema tabela stranica u dva nivoa



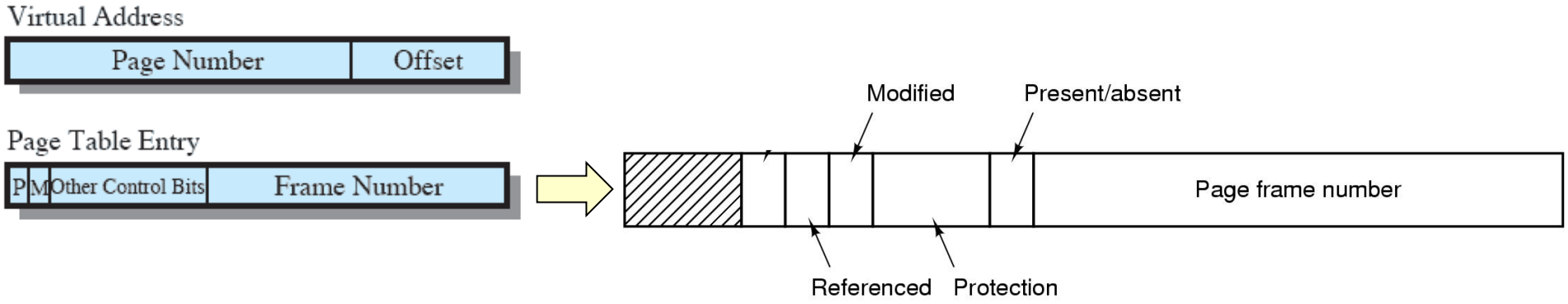
Tabele stranica u dva nivoa (3)

- Prevođenje virtuelne adrese



Struktura stavke u tabeli stranica

- Uobičajena veličina stavke (ulaza) iznosi 32 bita
  - Page Frame Number – broj fizičke stranice
  - Valid (*Present*) bit – ako je bit postavljen na 1, stranica je u memoriji; pristup stranici sa bitom vrednosti 0 generiše grešku zbog stranice (*page fault*)
  - Zaštita – prava pristupa sadržaju stranice (čitanje, upis, izvršavanje)
  - Bit modifikacije (*dirty*) – da li je stranica modifikovana u memoriji
  - Bit referenciranja (korišćenja) – postavlja se kad god je stranica referencirana; koristi se kod zamene stranica

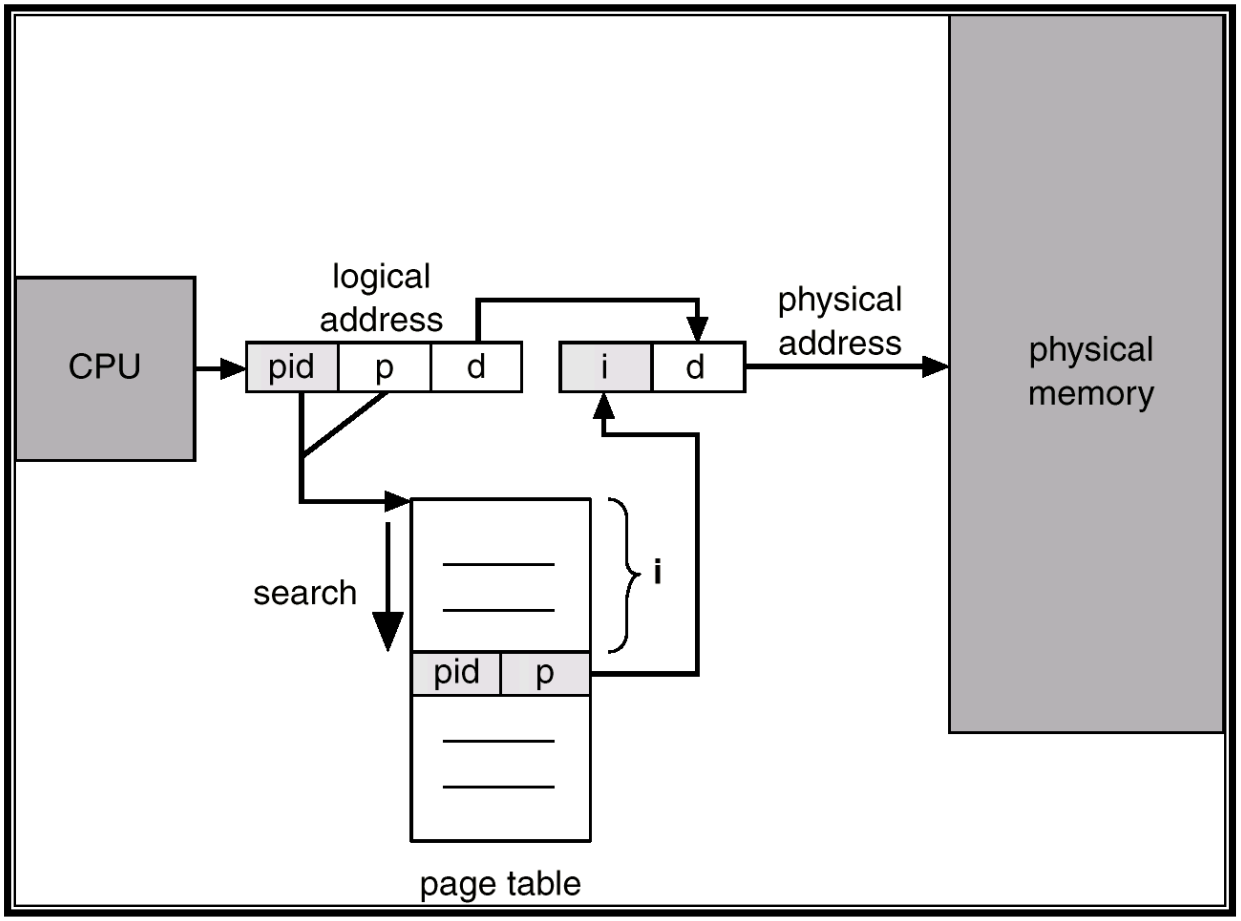


Invertovane tabele stranica (1)

- U tabeli stranica postoji jedna stavka za svaku fizičku stranicu (stranični okvir) u memoriji
- Svaka stavka u tabeli stranica sadrži virtuelnu adresu stranice koja je smeštena u datu fizičku stranicu, kao i podatke o procesu kojem ta stranica pripada
- Smanjuje memorijski prostor neophodan za smeštanje tabele stranica, ali povećava vreme neophodno za pretraživanje tabele po određenom broju virtuelne stranice
- Koristi se heš (hash) tabela, hešovana po virtuelnoj adresi za ubrzanje pretraživanja po prethodno opisanom principu
- Da bi se izbegao dodatno referenciranje memorije uz svaki memorijski pristup koristi se TLB
- Implementacija na PowerPC, UltraSPARC i IA-64

Invertovane tabele stranica (2)

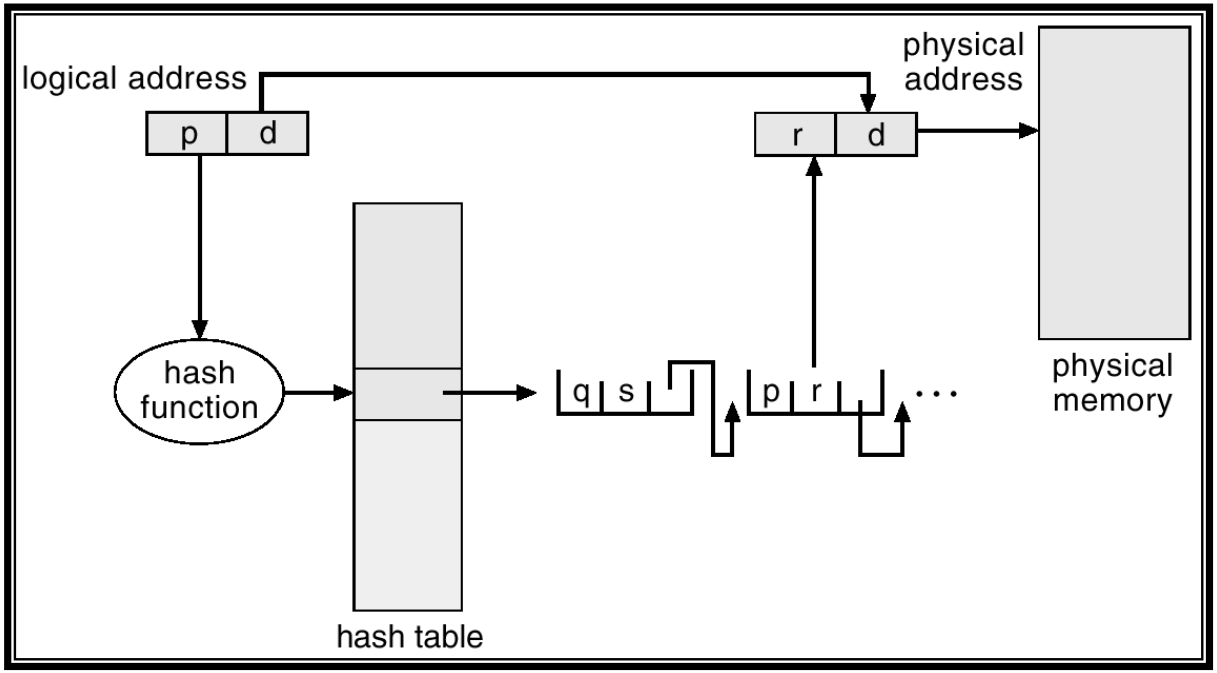
- Transformacija virtuelne (logičke) adrese u fizičku korišćenjem invertovane tabele stranica



Silberschatz, 2013

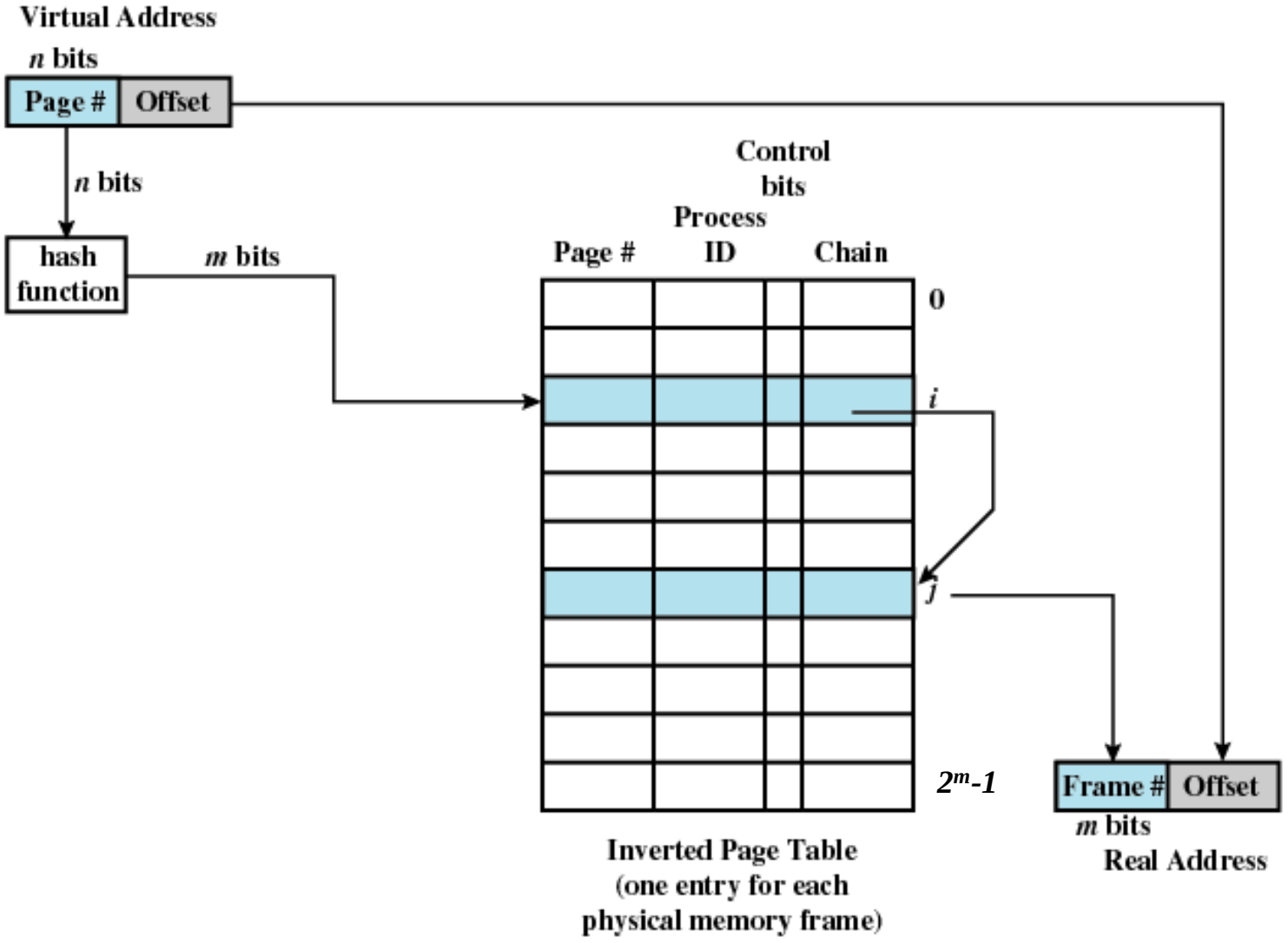
Invertovane tabele stranica sa heš funkcijom

- Broj virtuelne stranice se preslikava u heš vrednost korišćenjem heš funkcije. Heš vrednost predstavlja indeks (pokazivač) na invertovanu tabelu stranica čija svaka stavka sadrži lančanu listu elementa formata (virtuelne stranica, stranični okvir) koji su heš funkcijom preslikani u istu heš vrednost
- Na osnovu broja virtuelne stranice pretražuje se lista i određuje broj straničnog okvira



Silberschatz, 2013

Struktura invertovane tabele stranica



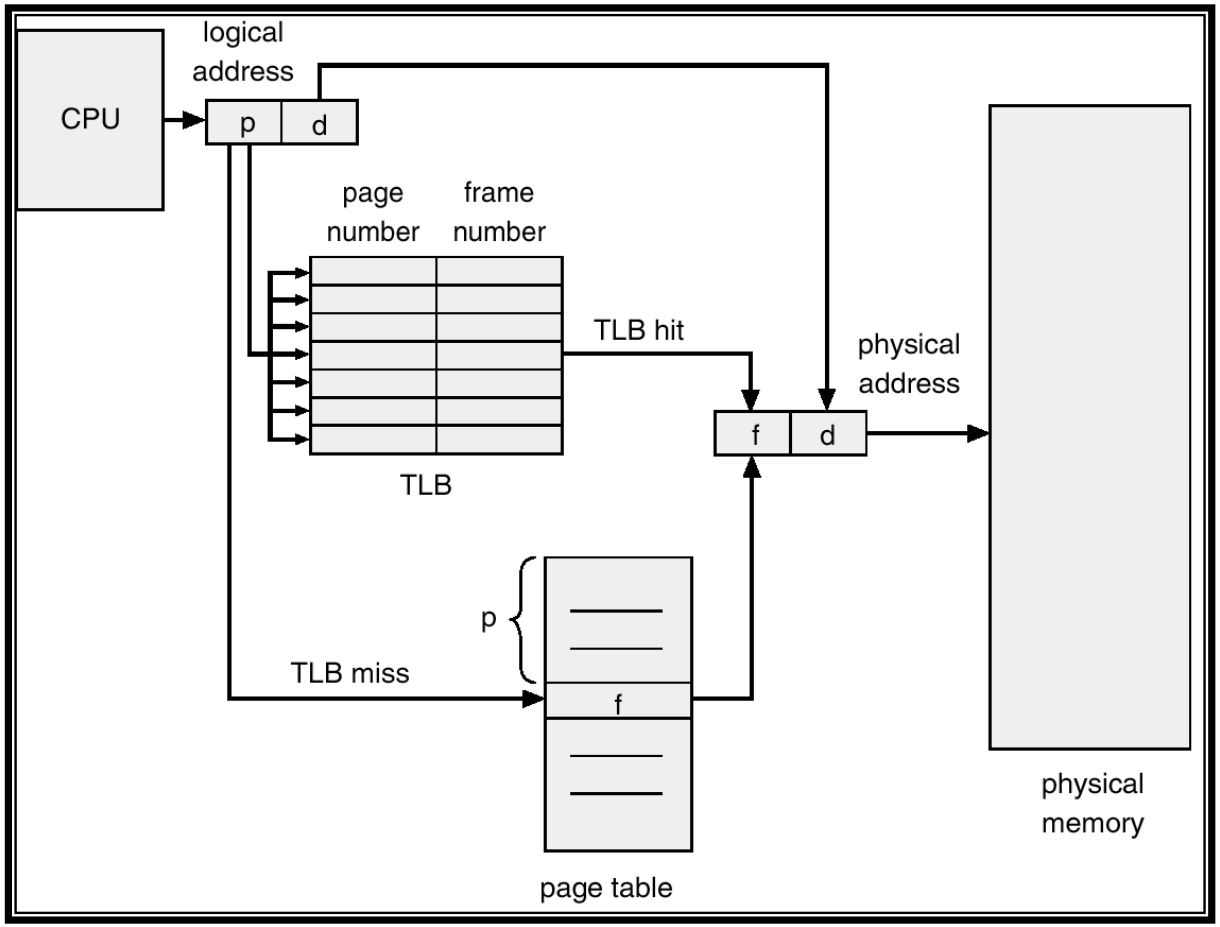
TLB – *Translation Lookaside Buffer*

- Hardverska komponenta u okviru MMU za transformisanje (prevođenje) virtuelnih u fizičke adrese bez pristupa tabeli stranica
- Asocijativna memorija –paralelno pretraživanje svih stavki u TLB-u po datom ključu (broju virtuelne stranice)
- Ima od 64 – 1024 stavki formata kao na sledećoj slici

Valid	Virtual page	Modified	Protection	Page frame
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Prevođenje adresa korišćenjem TLB

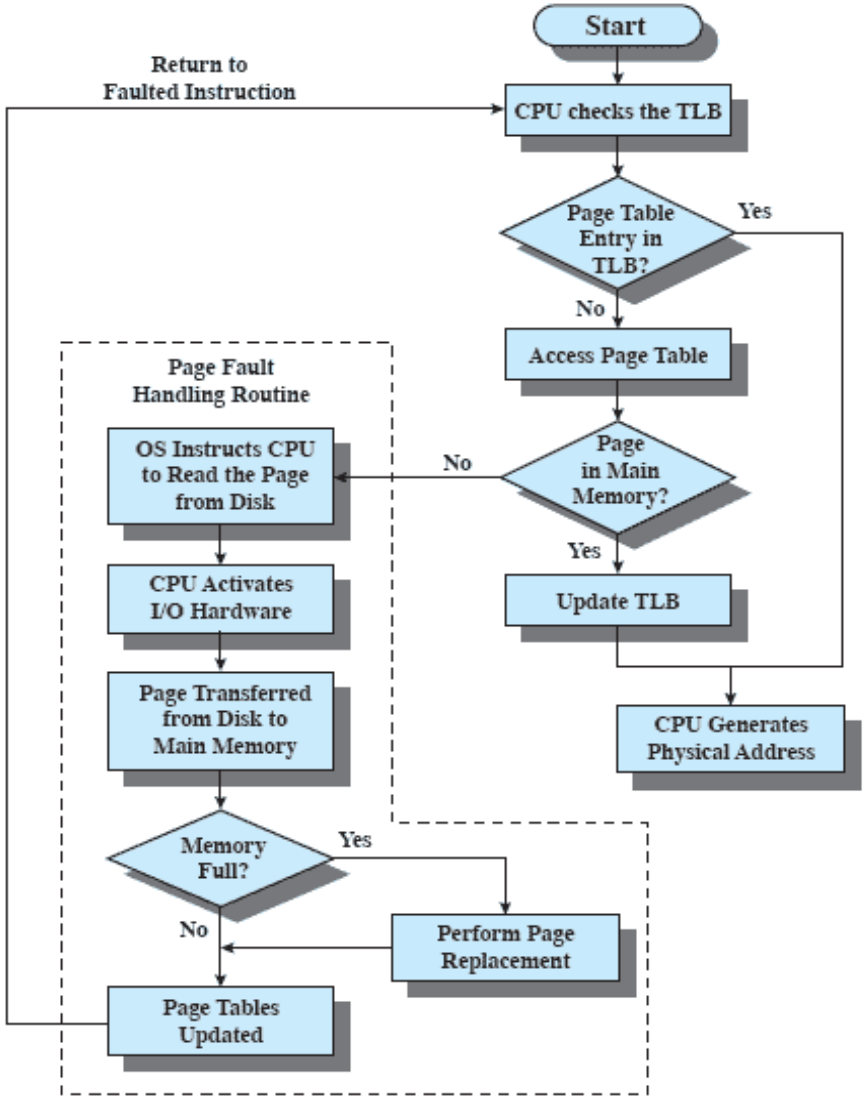
- Upravljanje TLB može biti hardversko od strane MMU, ili softversko od strane OS (RISC – SPARC, MIPS, Alpha, HP PA, itd.)



Silberschatz, 2013

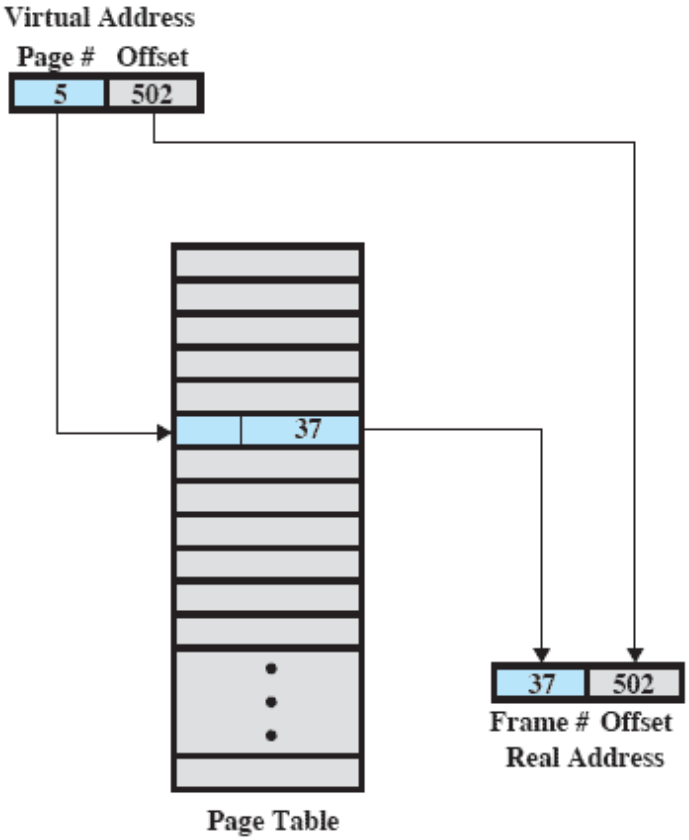
Algoritam prevođenja adresa

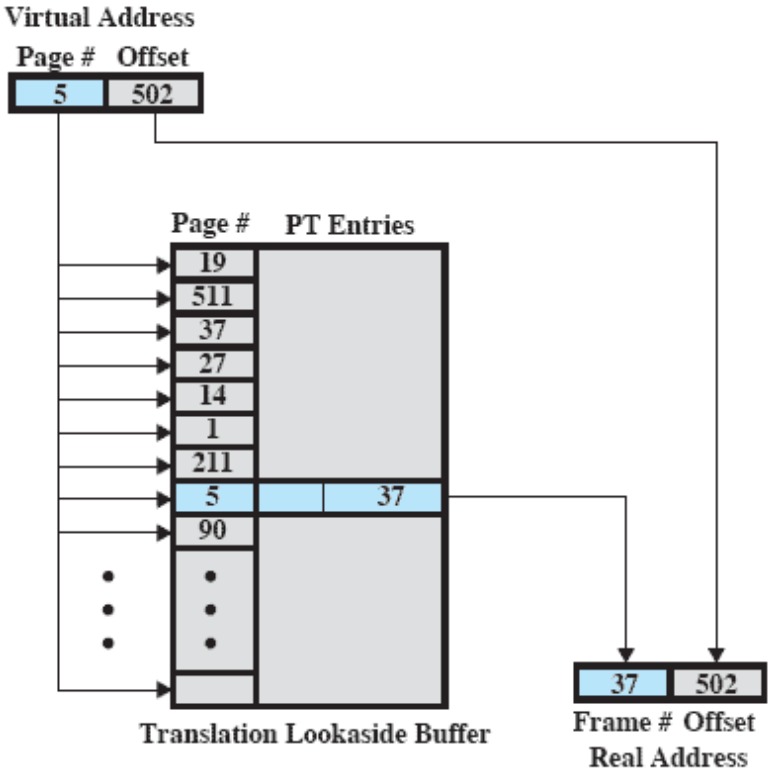
- Algoritam prevođenja adresa korišćenjem TLB i tabele stranica



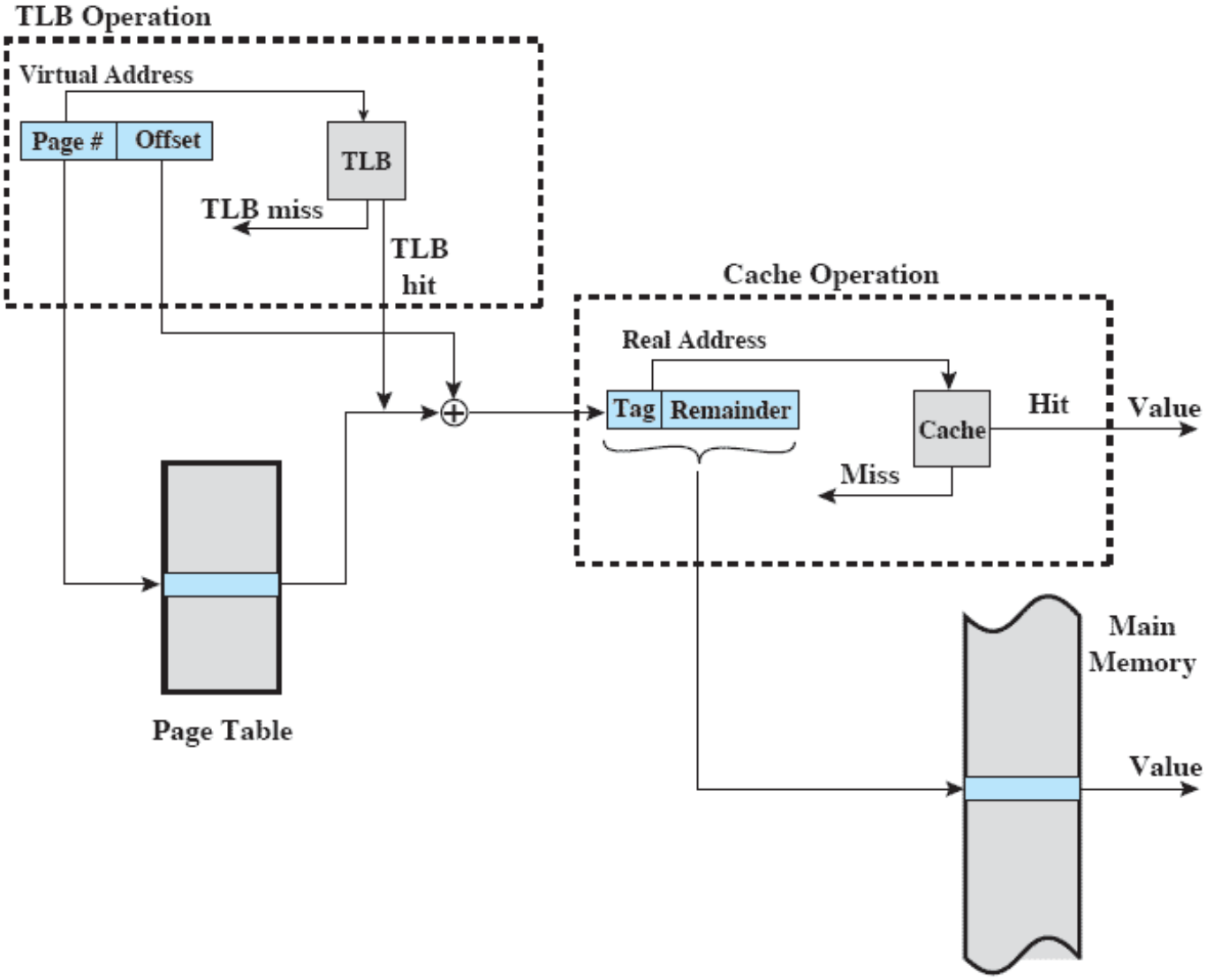
Direktno i asocijativno traženje

- Razlika između direktnog i asocijativnog traženja (lookup) stavke u tabeli stranica





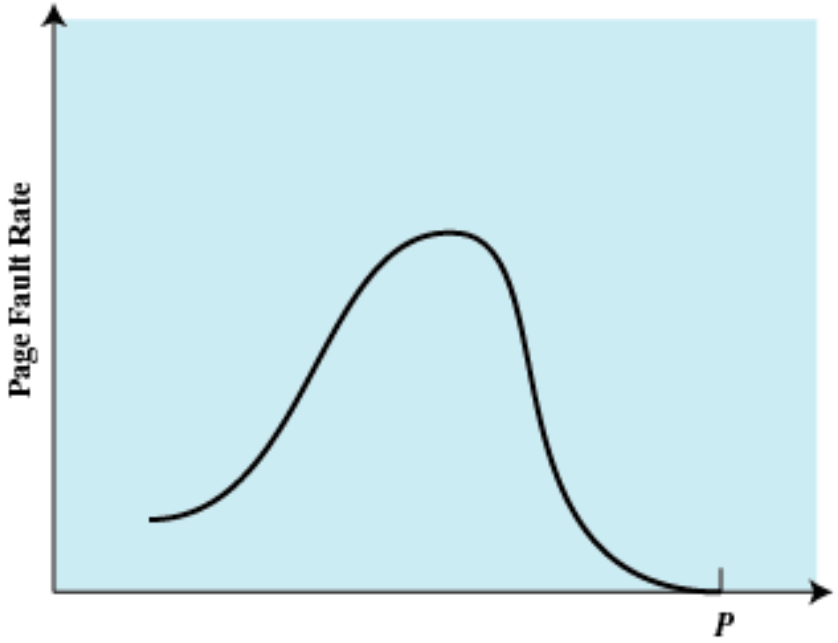
TLB i keš memorija



Veličina stranice

- Manja veličina stranice
  - Manja količina interne fragmentacije unutar poslednje stranice
  - Više stranica je neophodno za svaki proces, što znači veće tabele stranica
  - Veće tabele stranica znači da i tabele stranica moraju da se straniče, tj. da se formiraju u virtuelnoj memoriji, tako da za jednu memorijsku referencu može da postoji dvostruka greška stranica, jedna zbog tabele stranica, druga zbog same stranice
- Veće stranice
  - Sekundarna memorija je tako projektovana da efikasno prenosi velike blokove podataka tako da je prednost u većim stranicama radi efikasnijeg blokovskog prenosa
- Sa manjom veličinom stranice u memoriji može da bude veliki broj stranica procesa i relativno je mali broj grešaka stranice (*page fault*)
- Sa povećanjem veličine stranice gubi se princip lokalnosti i učestalost grešaka stranica raste, međutim na kraju učestalost opada sve do 0 kada je veličina stranice jednaka veličini procesa (ceo proces je u memoriji)

Veličina stranice (2)



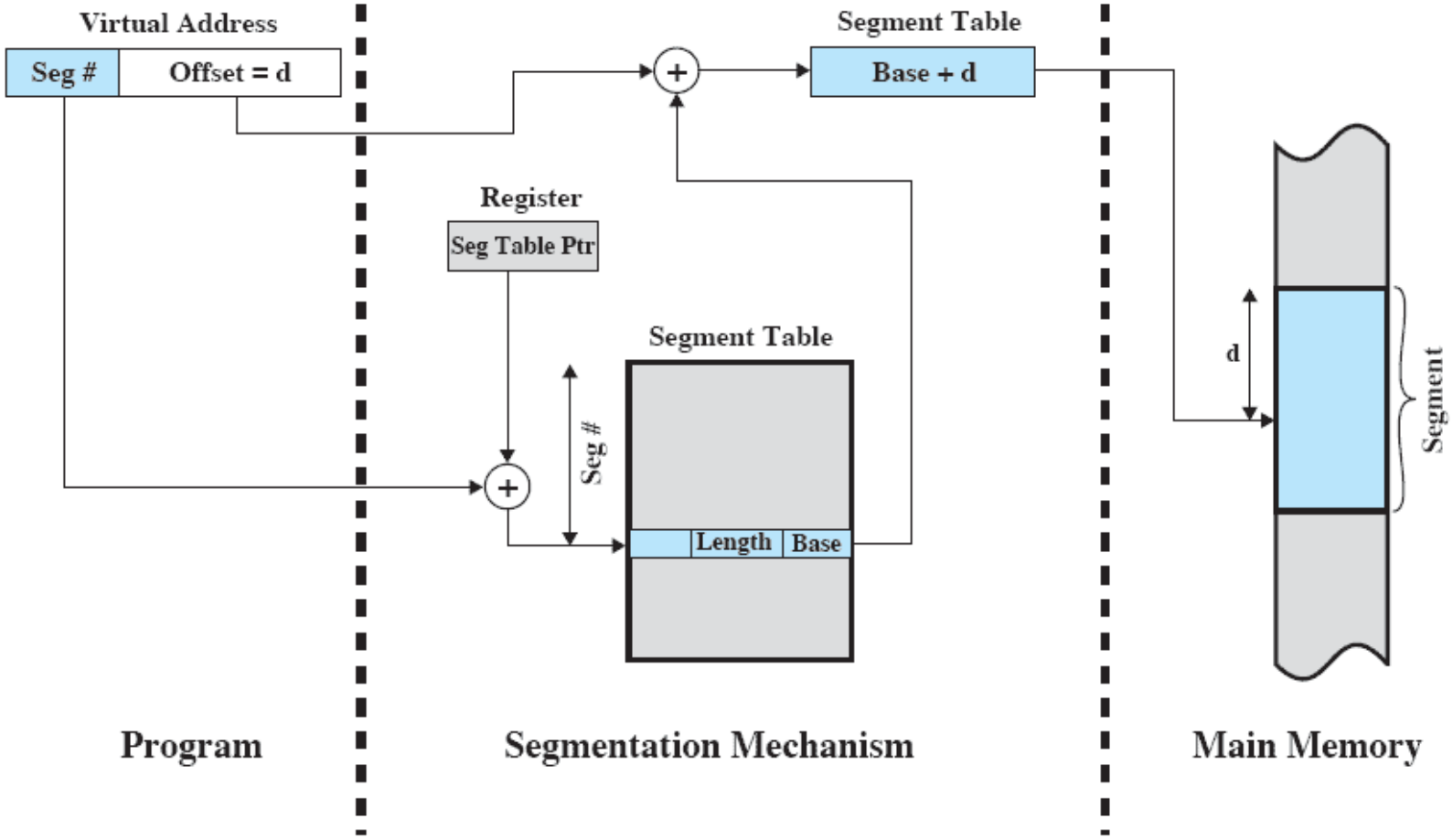
(a) Page Size

Computer	Page Size
Atlas	512 48-bit words
Honeywell-Multics	1,024 36-bit words
IBM 370/XA and 370/ESA	4 Kbytes
VAX family	512 bytes
IBM AS/400	512 bytes
DEC Alpha	8 Kbytes
MIPS	4 Kbytes to 16 Mbytes
UltraSPARC	8 Kbytes to 4 Mbytes
Pentium	4 Kbytes or 4 Mbytes
Intel Itanium	4 Kbytes to 256 Mbytes
Intel core i7	4 Kbytes to 1 Gbyte

P = size of entire process
W = working set size
N = total number of pages in process

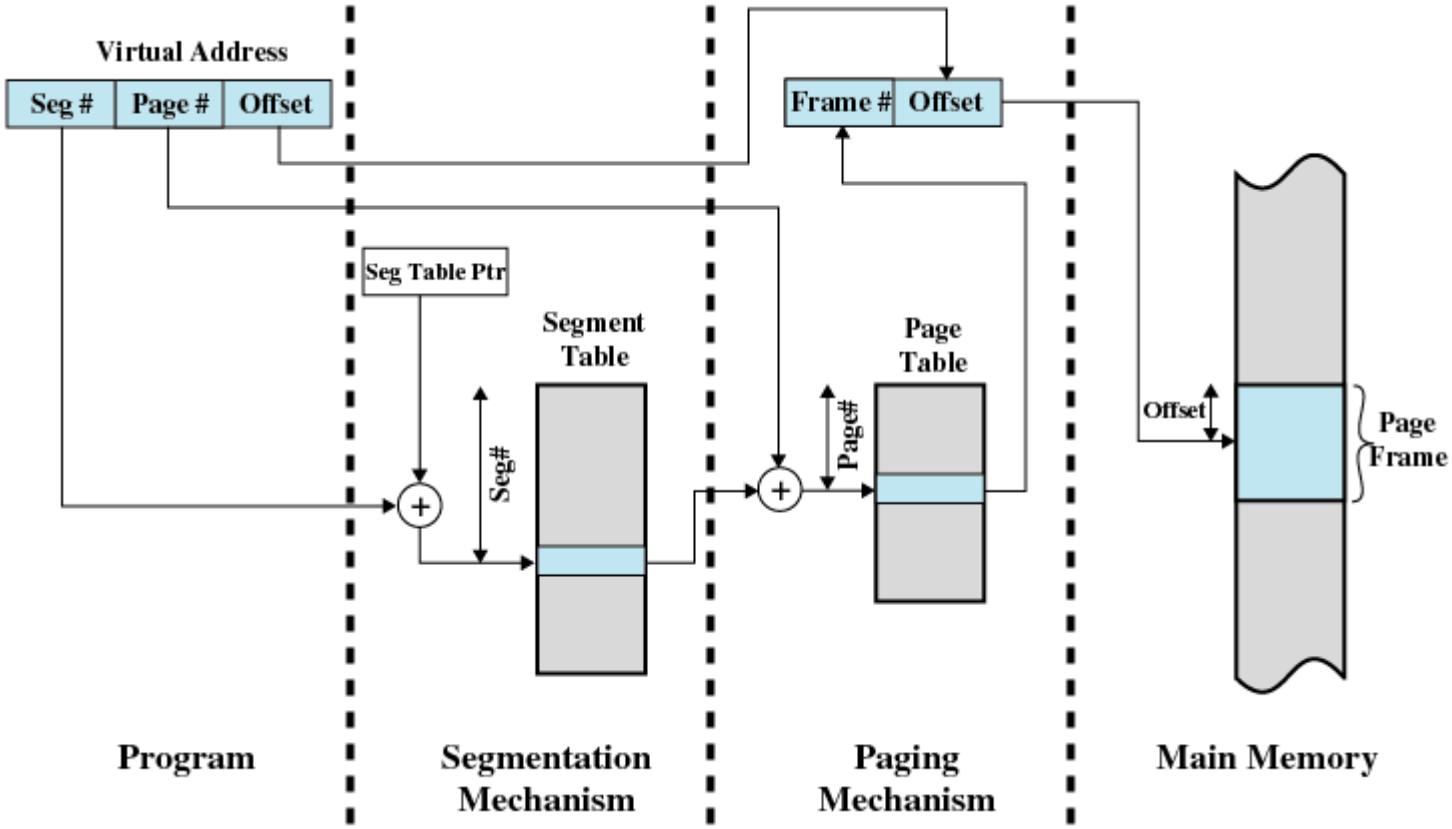
Segmentacija

- Prevođenje adrese u sistemu segmentacije



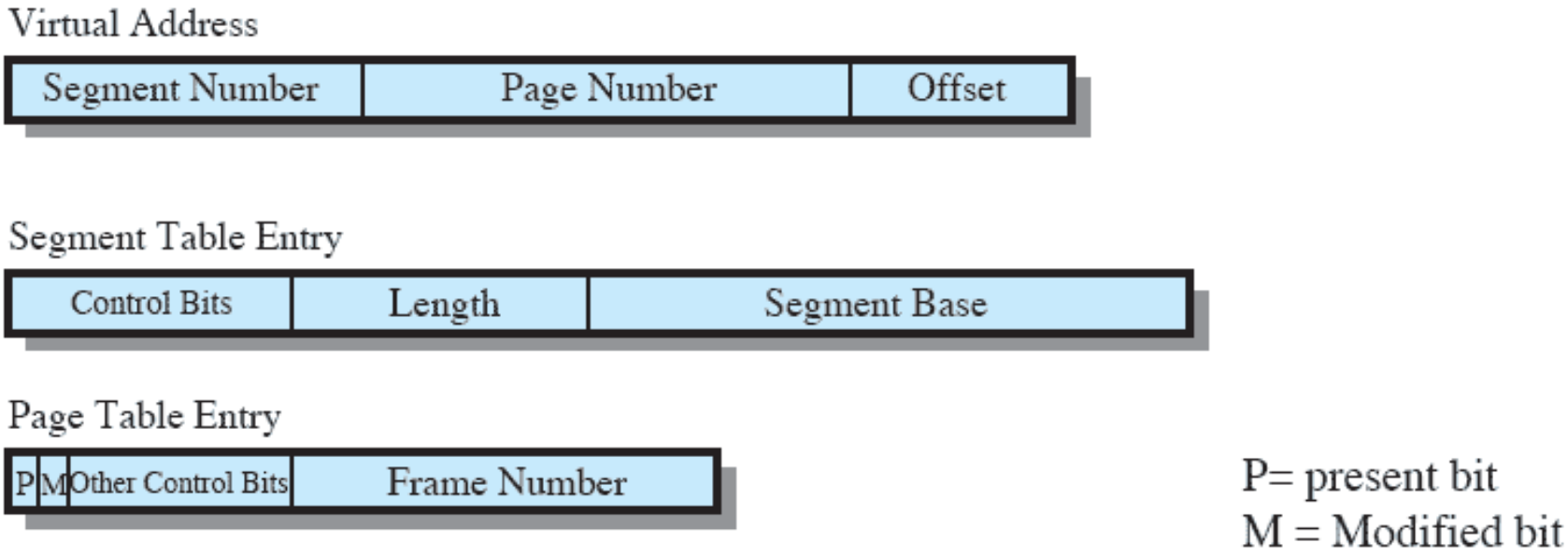
Segmentacija sa straničenjem

- Program je logički podeljen na segmente, a svaki segment na stranice
- Uključuje dobre karakteristike obe šeme upravljanja memorijom



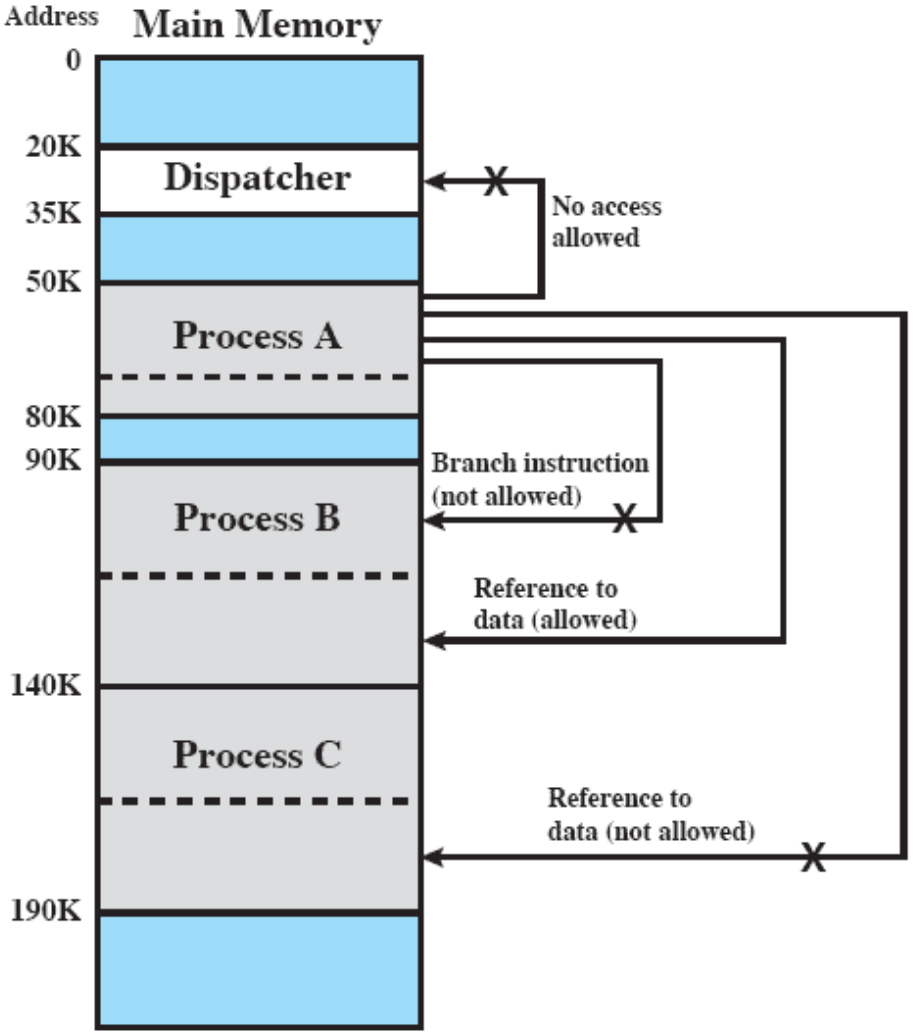
Segmentacija sa straničenjem (2)

- Virtuelna adresa, stavka tabele segmenata i stavka tabele stranica kod segmentcije sa straničenjem



Zaštita i deljenje memorije

- Mehanizmi zaštite segmenata



Virtuelna memorija

Softver operativnog sistema za upravljanje memorijom

- Sistem za upravljanje memorijom OS određen je sledećim:
  - Da li koristi tehnike virtuelne memroije
  - Da li koristi straničenje ili segmentaciju
  - Koje algoritme koristi za različite aspekte upravljanja memorijom

- Algoritmi (strategije) za upravljanje memorijom
  - Politika učitavanja (donošenja) stranice
  - Politika smeštanja stranice
  - Politika zamene
  - Upravljanje rezidentnim skupom
  - Politika čišćenja
  - Upravljanje učitavanjem

Politika učitavanja (donošenja)

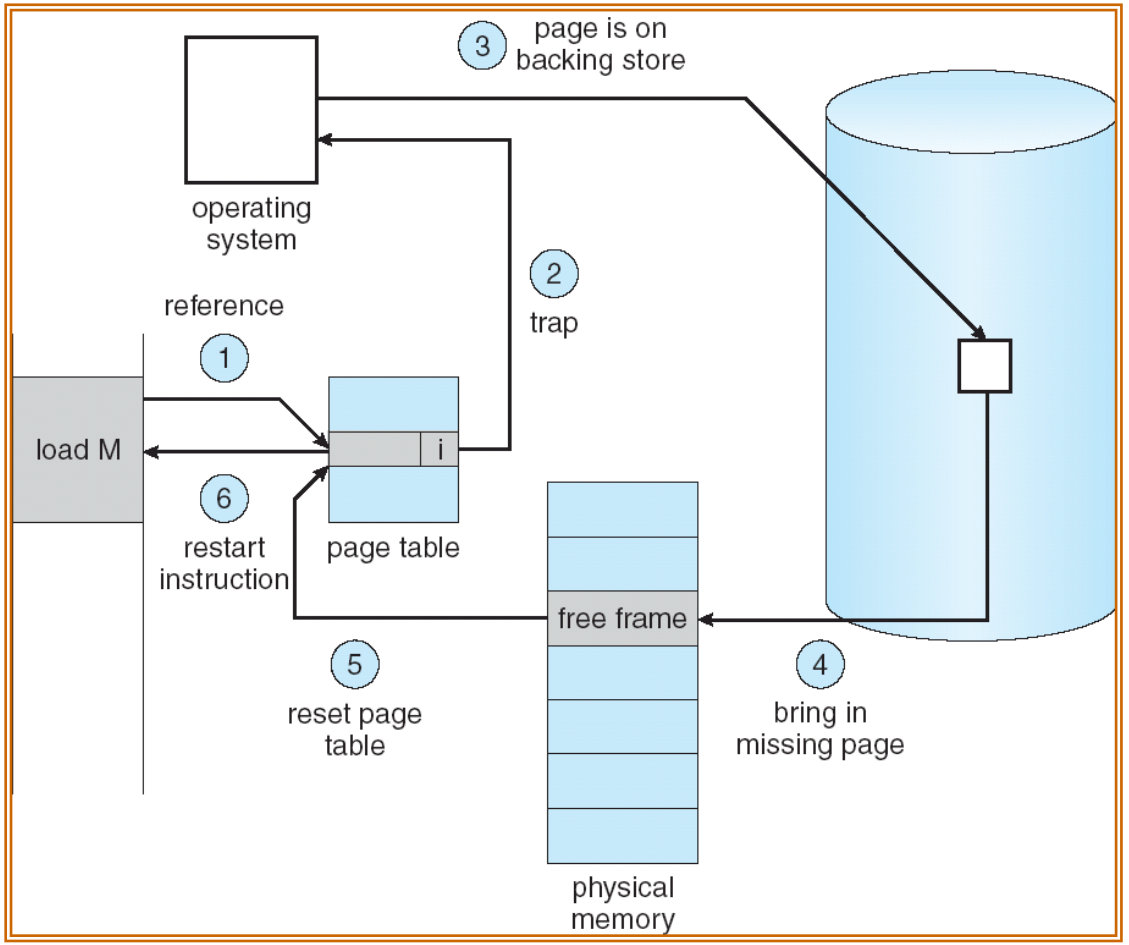
- Određuje kada stranica treba da bude učitana u glavnu memoriju
- Straničenje na zahtev (*demand paging*) učitava stranice u memoriju samo kada se napravi referenca na memorijsku lokaciju (adresu) unutar te stranice
  - Uzorkuje veliki broj grešaka stranice kada se proces startuje
- Prestraničenje (*Prepaging*) učitava u memoriju više stranica nego što je trenutno potrebno, ali se očekuje da će u bliskoj budućnosti biti
  - Efikasnije je učitati stranice koje su kontinualno smeštene na disku

Straničenje na zahtev

- Stranice se smeštaju (*load*) u glavnu memoriju tek kada se refrencira memorijska lokacija u okviru stranice
  - Na početku izvršavanja procesa generiše se veliki broj grešaka stranice (*page fault*)
- Kada se javi greška stranice, operativni sistem treba da stranicu učita sa diska, smesti u slobodni stranični okvir i ažurira tabelu stranica
- Ukoliko nema slobodnog straničnog okvira u memoriji, operativni sistem mora da odabere jednu stranicu koju će da obriše iz memorije i na njeno mesto smesti stranicu koja je generisala grešku - **Politika i algoritmi zamene stranica** (*Page replacement*) - generalni algoritmi primenljivi na keš memorije ili keš Web servera
- Ukoliko je stranica koja se briše modifikovana u memoriji, ona mora biti sačuvana na disku, ukoliko nije menjana (na primer, stranica sa kôdom programa), njena kopija na disku je ažurna i ona se samo prepisuje novom stranicom

Obrada greške stranica

1. Instrukcija referencira memorijsku lokaciju
2. Valid = 0 ->Trap u OS
3. Nalaženje odgovarajuće stranice na disku
4. Smeštanje stranice sa diska u memoriju (po potrebi osloboditi stranični okvir u memoriji)
5. Ažuriranje tabele stranica
6. Restartovanje instrukcije



Efektivno vreme pristupa memoriji

- p – verovatnoća nastanka greške stranica ($0 \leq p \leq 1$)
- Vreme za čuvanje stranja prekinutog procesa, obradu prekida i restartovanje prekinutog procesa je reda nekoliko 100 mikrosekundi do 1ms
- Vreme za zamenu stranica = rotation latency + seek + transfer
  = 8 ms + 15 ms + 1 ms = 24ms
- Vreme pristupa memoriji je 10ns – 200ns zavisno da li je sadržaj u kešu ili ne; srednje vreme 100ns
- **Efektivno vreme pristupa** = $(1 - p) * 100\ ns + p * (25\ ms + 100\ ns)$
  $= 100\ ns + p * 25\ 000\ 000\ ns$
- Ukoliko želimo da Efektivno vreme pristupa (EFP) bude do 10% lošije u odnosu na vreme pristupa bez grešaka stranica, mora se postići pojava najviše jedne greške stranica na $2.5 * 10^6$ pristupa memoriji

  EFP = 110 ns $\Longrightarrow$ $p = 1 / 2.5 * 10^6$

Politika smeštanja

- Određuje gde u stvarnoj memoriji proces treba da bude smešten (stranice procesa)
- U sistemu sa čistom segmentacijom politika smeštanja je važna zbog problema fragmentacije (najbolje poklapanje, prvo poklapanje, itd.)
- Za sistem koji koristi čisto straničenje ili segmentaciju sa straničenjem smeštanje je obično nevažno, jer je prevođenje virtuelnih adresa u fizičke podjednako efikasno za bilo koju kombinaciju stranica-okvir

Politika zamene

- Kada su svi stranični okviru u glavnoj memoriji zauzeti politika zamene određuje koju stranicu iz glavne memorije treba zameniti kad treba da se učita nova stranica
- Trebalo bi zameniti onu stranicu koja će sa najmanjom verovatnoćom biti referencirana u budućnosti
- Politike zamene pokušavaju da predvide buduće referenciranje stranica na osnovu referenciranja u prošlosti
- Zaključavanje straničnih okvira – stranice u zaključanim okvirima ne mogu da se zamene; okvir se zaključava na osnovu bita zaključavanja pridruženog svakom okviru
  - Kernel OS
  - Glavne upravljačke strukture
  - U/I baferi

Algoritmi zamene stranica

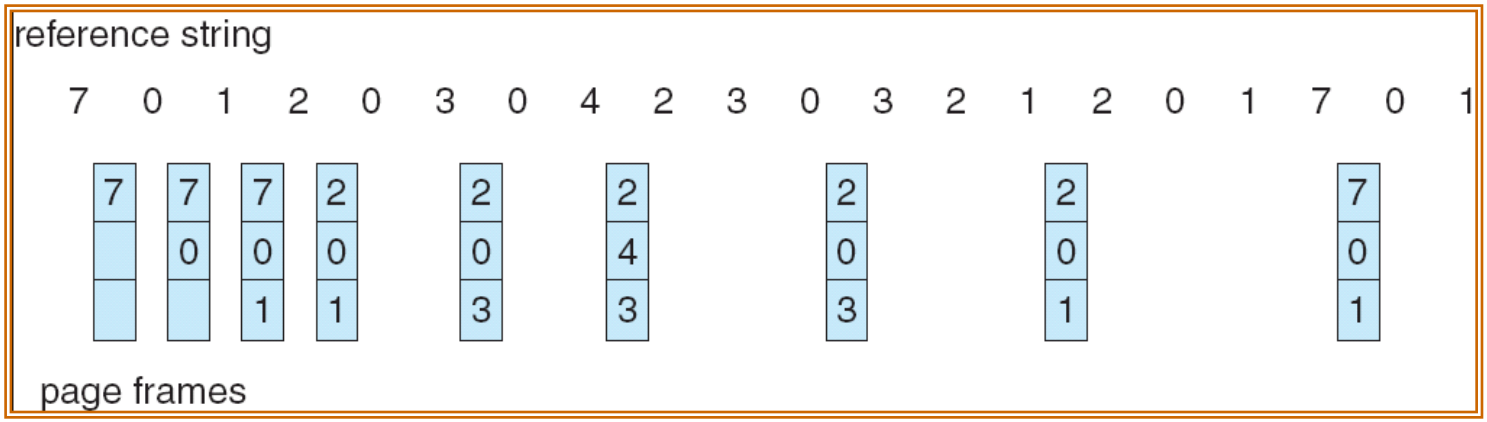
- Cilj je postizanje minimuma grešaka stranica
- Evaluacija algoritama na osnovu niza memorijskih referenci i izračunavanja broja grešaka stranica na tom nizu referenci
- Algoritmi:
  - Optimalni algoritam zamene stranica
  - Prva unutra, prva napolje (*FIFO – First-In, First-Out*)
  - Najmanje skoro korišćena stranica (*LRU – Least Recently Used*)
  - Algoritam časovnika (*Clock*)

Optimalni algoritam zamene stranica

- Najbolji algoritam, ali ga je nemoguće implementirati
- U trenutku nastanka greške stranica u memoriji su smeštene stranice različitih procesa; svaka od njih će (moguće) biti referencirana u budućnosti posle određenog broja instrukcija
- Za zamenu se bira ona stranica koja će najkasnije biti referencirana, tako da se greška stranica što je moguće kasnije odloži
- Implementacija je nemoguća, jer u trenutku greške stranica OS ne zna kada će koja stranica u memoriji biti referencirana
- Koristi se za evaluaciju algoritama zamene stranica; implementira se na osnovu prvog izvršenja programa i registrovanja referenci na stranice tokom izvršavanja

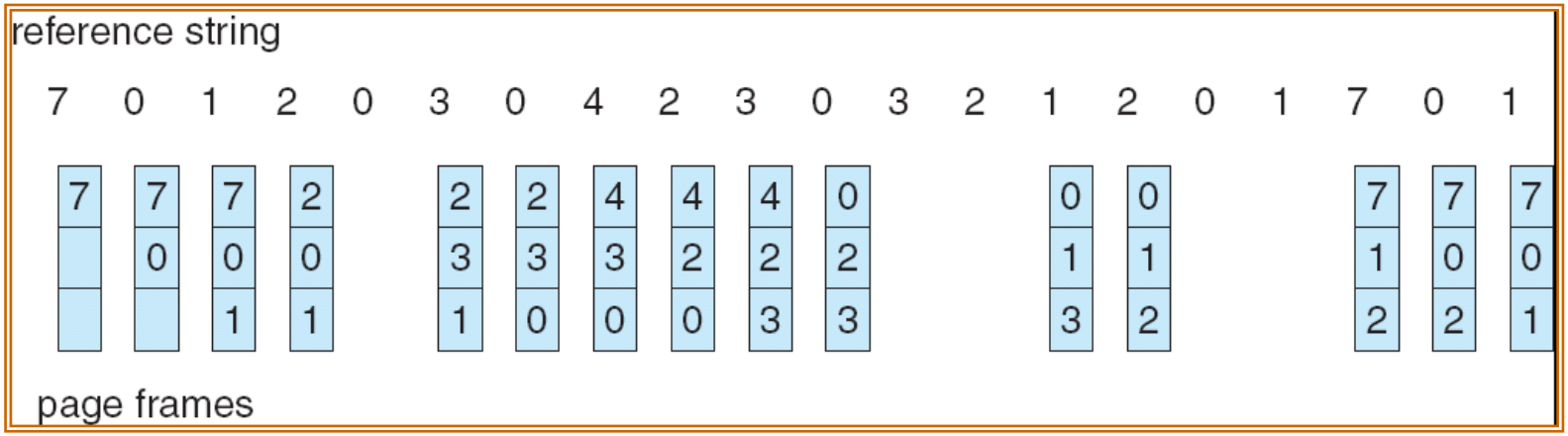
Optimalni algoritam - primer

- Evaluacija algoritma zamene stranica se izvodi njegovim izvršavanjem i primenom na određenoj nizu memorijskih referenci (*reference string*) i određivanjem broja grešaka stranica na tom nizu
  - Niz referenci: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
  - Memorija sadrži tri stranična okvira
- Generiše 9 grešaka stranica



FIFO algoritam

- Sistem za upravljanje memorijom održava listu svih stranica trenutno u memoriji, pri čemu je prva stranica u listi najranije učitana u memoriju ("najstarija"), a poslednja najskorije učitana ("najmlađa")
- Prilikom greške stranica, algoritam uklanja prvu stranicu u listi i zamenjuje je novom stranicom koju smešta u memoriju i dodaje na kraj liste
- FIFO algoritam se retko koristi u osnovnom obliku
- Primer
  - Algoritam generiše 15 grešaka stranica

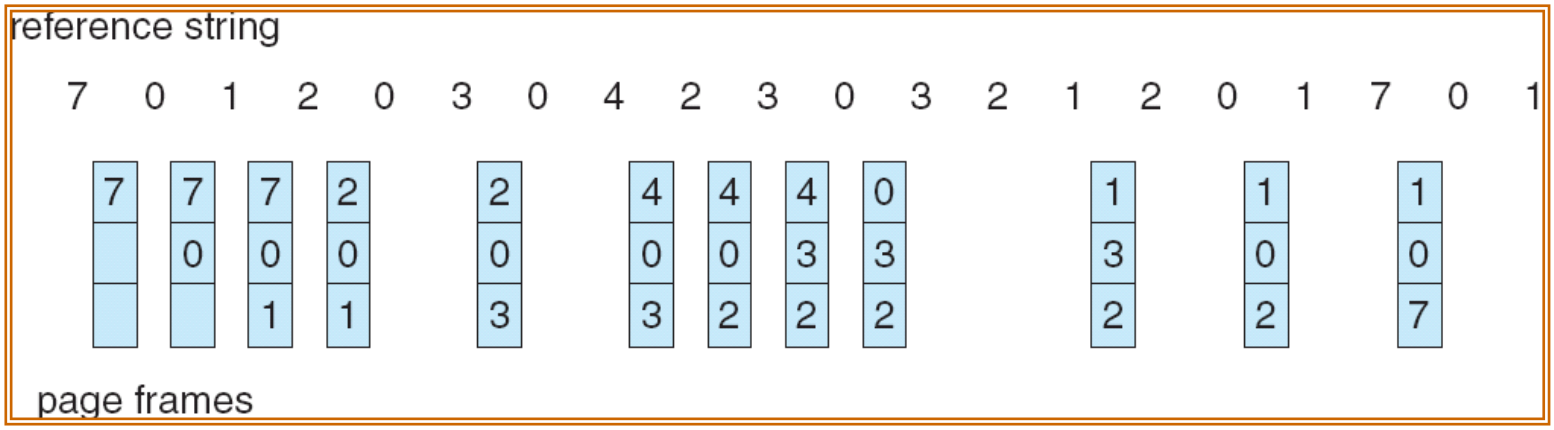


Algoritam zamene najmanje skoro korišćene stranice (LRU)

- Zasniva se na pretpostavci da stranice koje su poslednjih nekoliko instrukcija bile referencirane, biće referencirane i u narednim instrukcijama
- Prilikom greške stranica, zamenjuje se stranica koja najduže vreme nije referencirana.
- Težak za implementaciju - Uz svaku stranicu u memoriji registruje se vreme poslednje reference te stranice što bi uzrokovalo velike režijske troškove za evidentiranje i pretraživanje vremena referenciranja

LRU algoritam - primer

- Za test niz referenci generiše 12 grešaka stranica

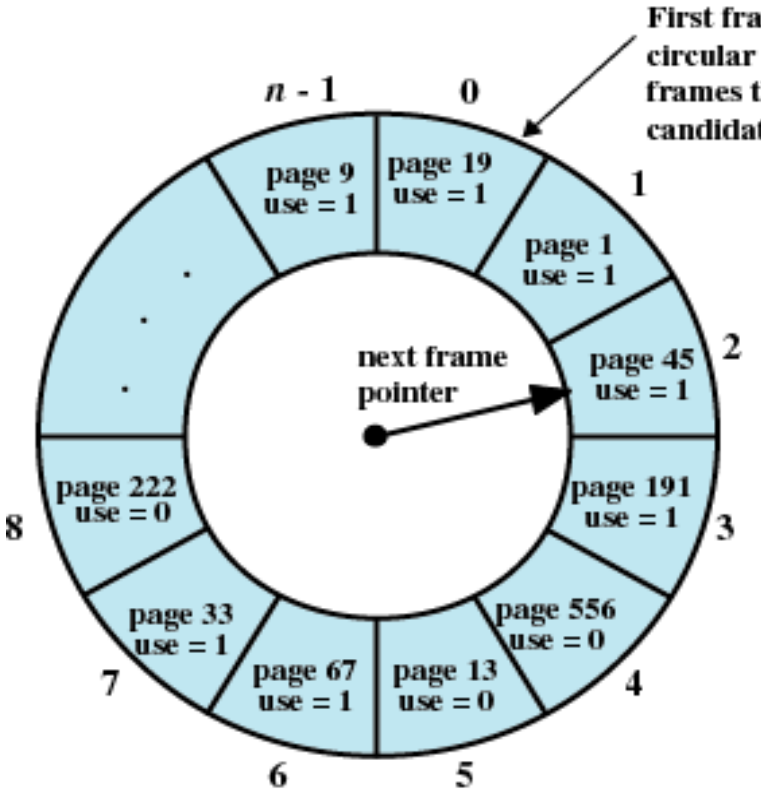


Algoritam časovnika (*Clock*)

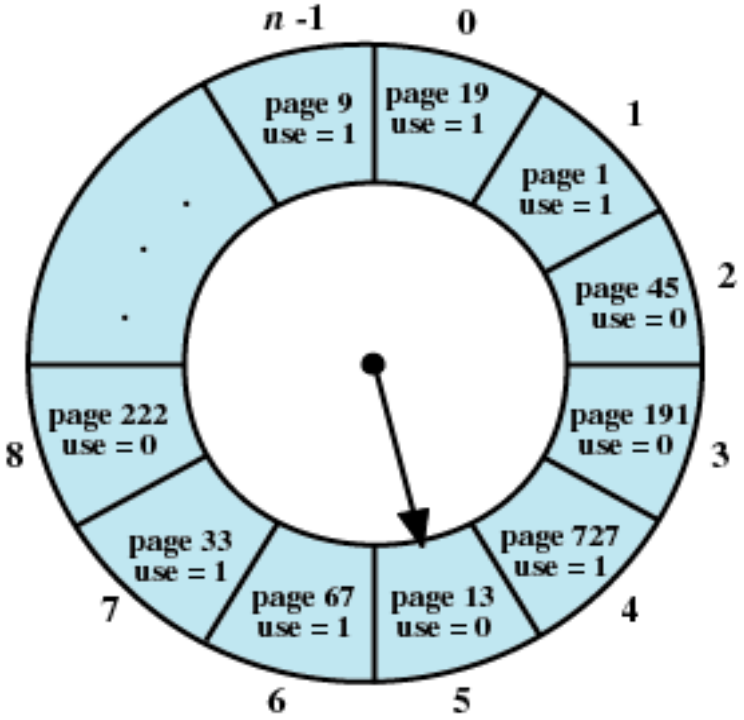
- Jednostavna modifikacija FIFO algoritma koja pored "starosti" stranice uzima u obzir i vrednost bita upotrebe (*use*) – ***u*** bita, koji se postavlja se na 1 kada se stranica učita u memoriju i kad god je referencirana
- Algoritam časovnika posmatra skup svih okvira koji su kandidati za zamenu (ako se radi o okvirima koje zauzima taj proces – lokalni opseg, a svih okvira u glavnoj memoriji – globalni opseg) kao kružni bafer kome je pridružen pokazivač
  - Ako je ***u*** = 0, stranica je i stara i ne skoro referencirana pa se zamenjuje novom stranicom
  - Ako je ***u*** = 1, bit se postavlja na 0, pokazivač prelazi na sledeći okvir u kružnom baferu i ispitivanje stranica se nastavlja
- Ukoliko su ***u*** bitovi svih stranica postavljeni na 1, algoritam postaje FIFO

Algoritam časovnika

- Kružni bafer od n okvira, zahteva se učitavanje stranice 727



(a) State of buffer just prior to a page replacement



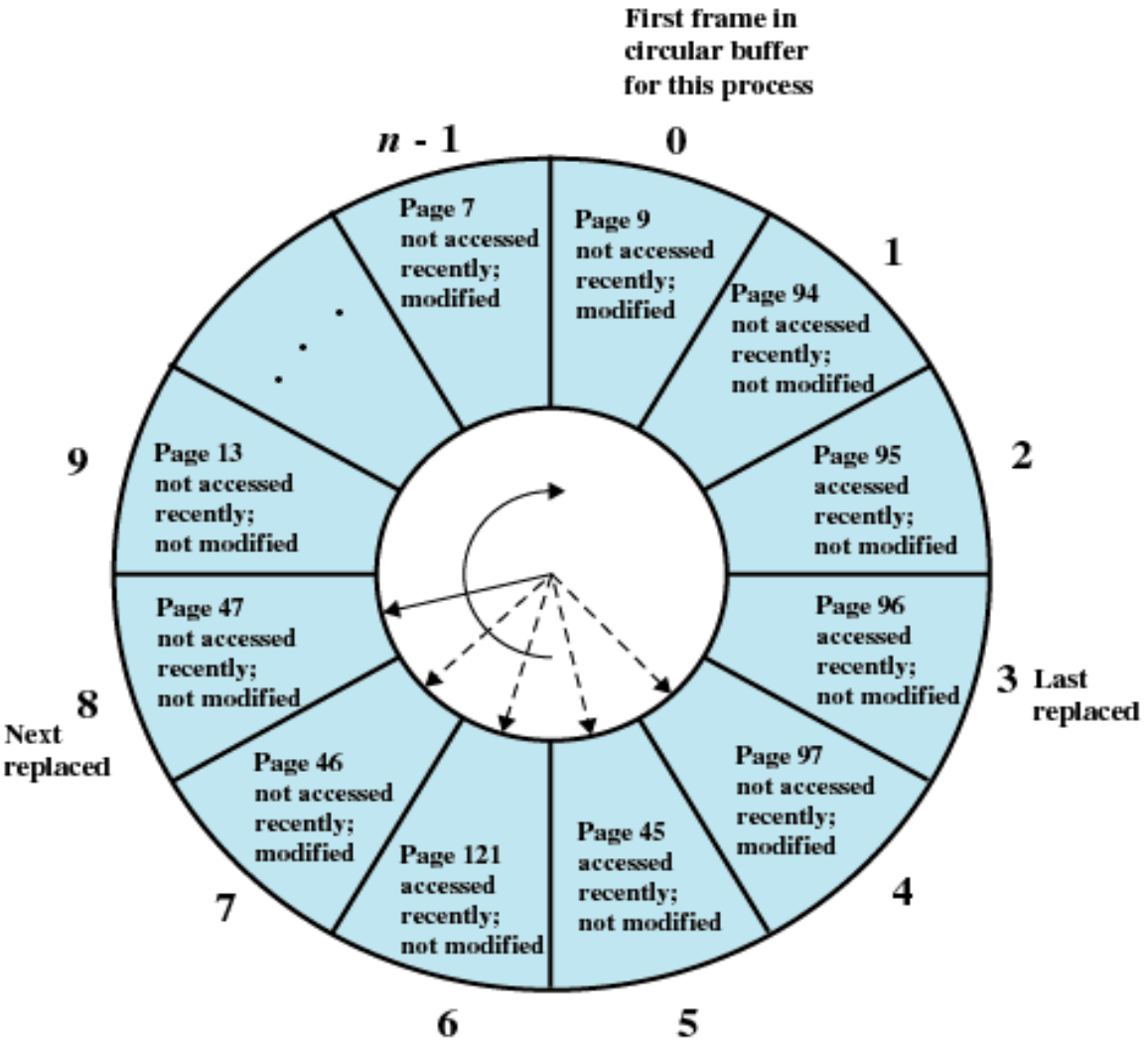
(b) State of buffer just after the next page replacement

Modifikacija algoritma časovnika

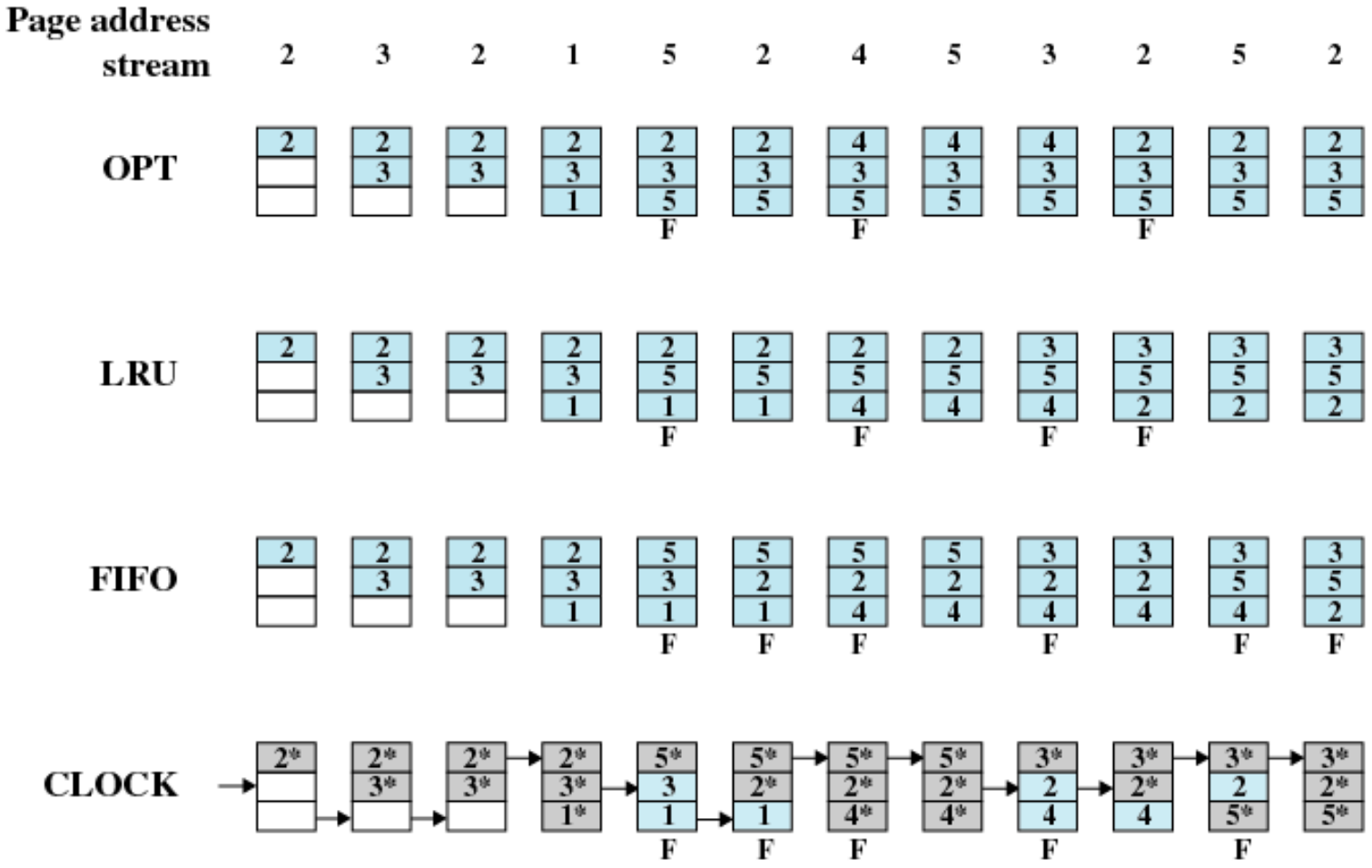
- Koristi se još jedan bit pridružen svakoj stranici
  - Bit promene (***m*** bit) – postavlja se na 1 kada je stranica modifikovana
- Ovaj bit je potreban da kada se sadržaj stranice promeni ona ne bude zamenjena dok se ne upiše na sekundarnu memoriju, a pošto je to vremenski "skupa" operacija, onda je bolje zameniti stranicu koja nije modifikovana
- Četiri klase stranica
  - Stranica nije referencirana niti modifikovana - (***u***=0, ***m***=0)
  - Stranica nije referencirana, ali je modifikovana - (***u***=0, ***m***=1)
  - Stranica je referencirana, ali nije modifikovana - (***u***=1, ***m***=0)
  - Stranica je referencirana i modifikovana - (***u***=1, ***m***=1)
- Algoritam
  1. Algoritam obilazi bafer okvira počev od pozicije pokazivača, pri čemu ne menja bit upotrebe stranicama kod kojih je ***u***=1. Bira za zamenu prvi okvir sa vrednostima ***u***=0 i ***m***=0 za stranicu u njemu
  2. Ako ne uspe u prvom prolazu, ponovo obilazi bafer okvira i traži okvir sa vrednostima ***u***=0 i ***m***=1 i bira za zamenu prvi takav okvir na koji naiđe. U toku skeniranja postavlja bit upotrebe na 0 u svakom okviru koji obiđe
  3. Ako ne uspe korak 2, pokazivač je vraćen na prvobitni položaj i svi okviri u skupu će imati bit upotrebe 0, pa ponavlja korak 1

Modifikacija algoritma časovnika (2)

- Primer



Algoritmi zamene stranica – primer 2



Baferovanje stranica

- Stranica koja je izabrana za zamenu se ne izbacuje iz memorije (briše) već se stavlja u jednu od dve liste:
  - Listu slobodnih stranica ukoliko stranica nije menjana
  - Listu modifikovanih stranica ukoliko jeste
- Sadržaj stranice ostaje u memoriji, a uklanja se samo stavka za tu stranicu u tabeli stranica (ili postavlja bit prisutnosti P na 0)
  - Grupišu se modifikovane stranice, pa se odjednom upisuju na disk čime se redukuje broj U/I operacija i poboljšavaju performanse
- Operativni sistem u svakom trenutku održava određeni broj slobodnih okvira – kada nova stranica treba da se učita (zbog greške stranice) bira se prvi okvir/stranica iz liste slobodnih stranica i novom stranicom prebriše prethodni sadržaj
- Stranica koja treba da se zameni ostaje u memoriji, pa ako je proces opet referencira ona se vraća u rezidentni skup tog procesa

Upravljanje rezidentnim skupom

- Veličina rezidentnog skupa – koliko stranica procesa učitati u memoriju
  - Fiksno dodeljivanje
    - Dodeljuje procesu fiksan broj okvira u glavnoj memoriji unutar kojih se izvršava
  - Promenljivo dodeljivanje
    - Dozvoljava da se broj okvira dodeljenih procesu, a samim tim i njegov rezidentni skup menja tokom vremena
- Opseg zamene – čiju stranicu zameniti kada dođe do greške stranice
  - Lokalna zamena
    - Bira se stranica za zamenu samo između stranica procesa koji je napravio grešku stranice
  - Globalna zamena
    - Razmatra sve nezaključane stranice u memoriji kao kandidate za zamenu

Upravljanje rezidentnim skupom (2)

- Moguće strategije upravljanja rezidentnim skupom
  - Fiksno dodeljivanje, lokalni opseg
  - Promenljivo dodeljivanje, lokalni opseg
  - Promenljivo dodeljivanje, globalni opseg
    - Implementirano od strane mnogih operativnih sistema

Strategija radnog skupa (*Working Set*)

- Tokom određene faze u izvršenju, proces referencira relativno mali deo svojih stranica – lokalnost referenci
- Radni skup - skup stranica koje proces trenutno koristi
- Radni skup W(t, Δ) u virtuelnom trenutku t čine stranice referencirane u poslednjih Δ jedinica virtuelnog vremena (memorijskih referenci)
  - Virtuelno vreme se meri u memorijskim referencama
  - Δ – okvir (prozor) virtuelnog vremena u kome se proces posmatra
- Strategija
  - Nadgledati radni skup svakog procesa
  - Periodično uklanjati iz rezidentnog skupa procesa one stranice koje nisu u radnom skupu (koristi se LRU)
  - Ukoliko je celokupni radni skup u memoriji, proces se izvršava bez greški stranica, sve dok ne pređe u sledeću fazu izvršenja
  - Prilikom greške stranica treba naći stranicu koja nije u radnom skupu procesa i zameniti je novom stranicom u memoriji

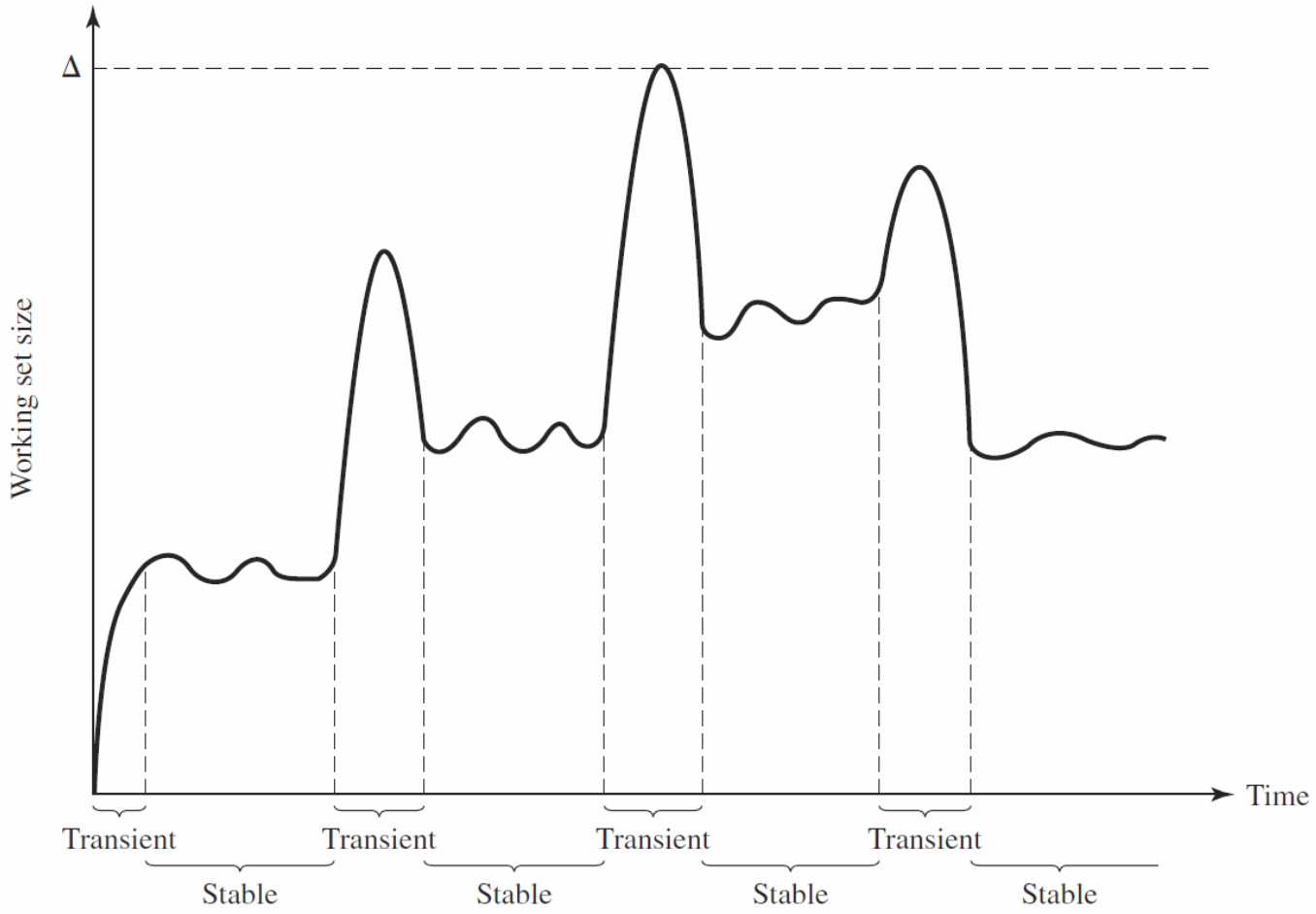
Radni skup procesa

- Radni skup procesa u zavisnosti od veličine "prozora"

Sequence of Page References	Window Size, Δ			
	2	3	4	5
24	24	24	24	24
15	24 15	24 15	24 15	24 15
18	15 18	24 15 18	24 15 18	24 15 18
23	18 23	15 18 23	24 15 18 23	24 15 18 23
24	23 24	18 23 24	•	•
17	24 17	23 24 17	18 23 24 17	15 18 23 24 17
18	17 18	24 17 18	•	18 23 24 17
24	18 24	•	24 17 18	•
18	•	18 24	•	24 17 18
17	18 17	24 18 17	•	•
17	17	18 17	•	•
15	17 15	17 15	18 17 15	24 18 17 15
24	15 24	17 15 24	17 15 24	•
17	24 17	•	•	17 15 24
24	•	24 17	•	•
18	24 18	17 24 18	17 24 18	15 17 24 18

Veličina radnog skupa

- Graf promene veličine radnog skupa tokom izvršenja procesa



Politika čišćenja

- Određuje kada promenjena stranica treba da bude upisana na sekundarnu memoriju
- Dve tehnike
  - Čišćenje na zahtev – stranica se upisuje na sekundarnu memoriju samo kada se izabere za zamenu
  - Predčišćenje – promenjene stranice se upisuju pre nego što su njihovi stranični okviri potrebni
- Najbolje je koristiti **baferovanje stranica** – zamenjene stranice se smeštaju na dve liste: one koje su promenjene i one koje nisu. Stranice na listi promenjenih mogu periodično da se upisuju na sekundarnu memoriju u paketima i pomeraju na listu nepromenjenih. Stranica na listi nepromenjenih se ili ponovo uspostavlja kada se referencira, ili se gubi kada se njen okvir dodeli novoj stranici

Upravljanje učitavanjem

- Određuje broj procesa koji će biti rezidentni u glavnoj memoriji (čije su stranice smeštene u glavnu memoriju) – nivo multiprogramiranja
- Ako je suviše malo procesa rezidentno u bilo kom trenutku, često će biti situacija da su svi blokirani i CPU nezaposlen, i trošiće se mnogo vremena na razmenu sadržaja između glavne i sekundarne memorije (*swapping*)
- Ako je suviše procesa rezidentno u memoriji, njihove stranice koje su učitane u memoriju uglavnom neće predstavljati radni skup, tako da će dolaziti do velike učestanosti grešaka stranica (*trashing*)
- Ako nivo multiprogramiranja treba da se smanji, jedan ili više procesa treba da se suspenduje (njegove stranice swap-uju na sekundarnu memoriju)

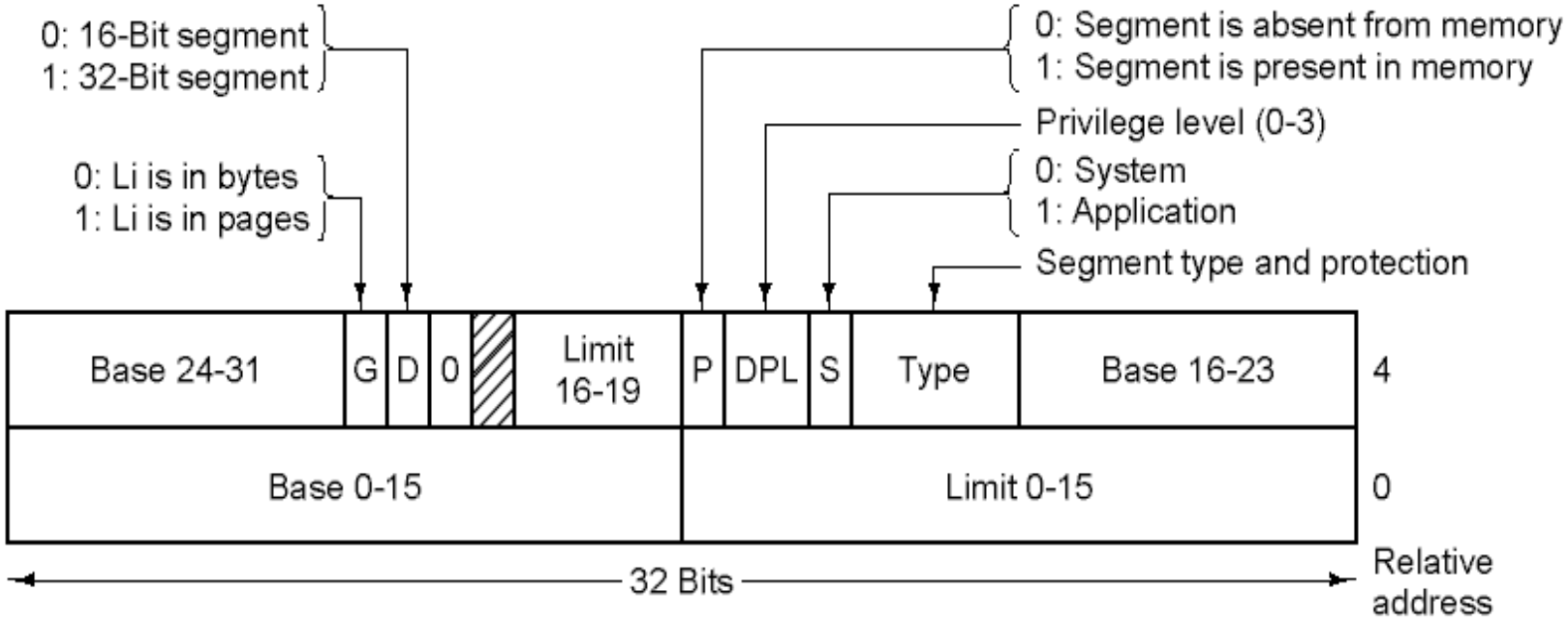
Upravljanje memorijom u realnim operativnim sistemima

Intel Pentium

- Segmentacija sa straničenjem (stranice veličine 4KB)
- Pentium obezbeđuje programima virtuelni adresni prostor od 16K nezavisnih segmenata, svaki sa 1G 32-bitnih reči
- Osnovu virtuelne memorije Pentiuma čine dve tabele
  - LDT (Local Descriptor Table) – svaki program ima sopstvenu tabelu i ona opisuje segmente programa: kod, podatke, magacin, itd.
  - GDT (Global Descriptor Table) – jedinstvena tabela koju dele svi programi – opisuje sistemske segmente uključujući i OS
- Da bi se pristupilo segmentu, selektor segmenta mora biti smešten u jedan od 6 segmentnih registara procesora
- Svaki selektor je 16 bitova
  - Indeks u LDT/GDT (13 bitova) – svaka tabela sadrži maksimum 8K deskriptora segmenata
  - Prava pristupa i zaštita (2 bita)
  - Da li se radi o lokalnom (LDT) ili globalnom (GDT) segmentu (1 bit)

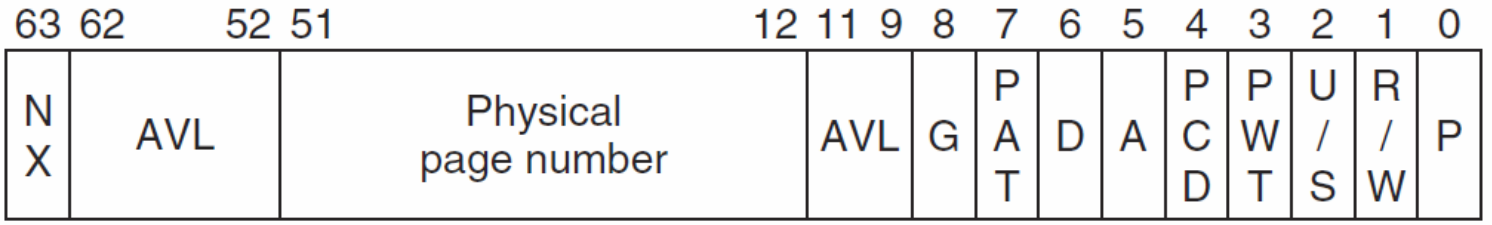
Pentium – Deskriptor segmenta

- Deskriptor segmenta je veličine 8 B:
  - Bazna adresa segmenta (Base) (32 b)
  - Veličina segmenta (Limit) (20 b) u bajtovima ili stranicama (1 b)
  - Tip i zaštita segmenta, sistemski/korisnički segment (1 b), nivo privilegija (2 b), segment u memoriji (1 b), itd.



Stavka tabele stranica – Intel procesori

- Stavka tabele stranica - Page Table Entry (PTE) –
  - MS Windows na Intel x86 i AMD x64 arhitekturama

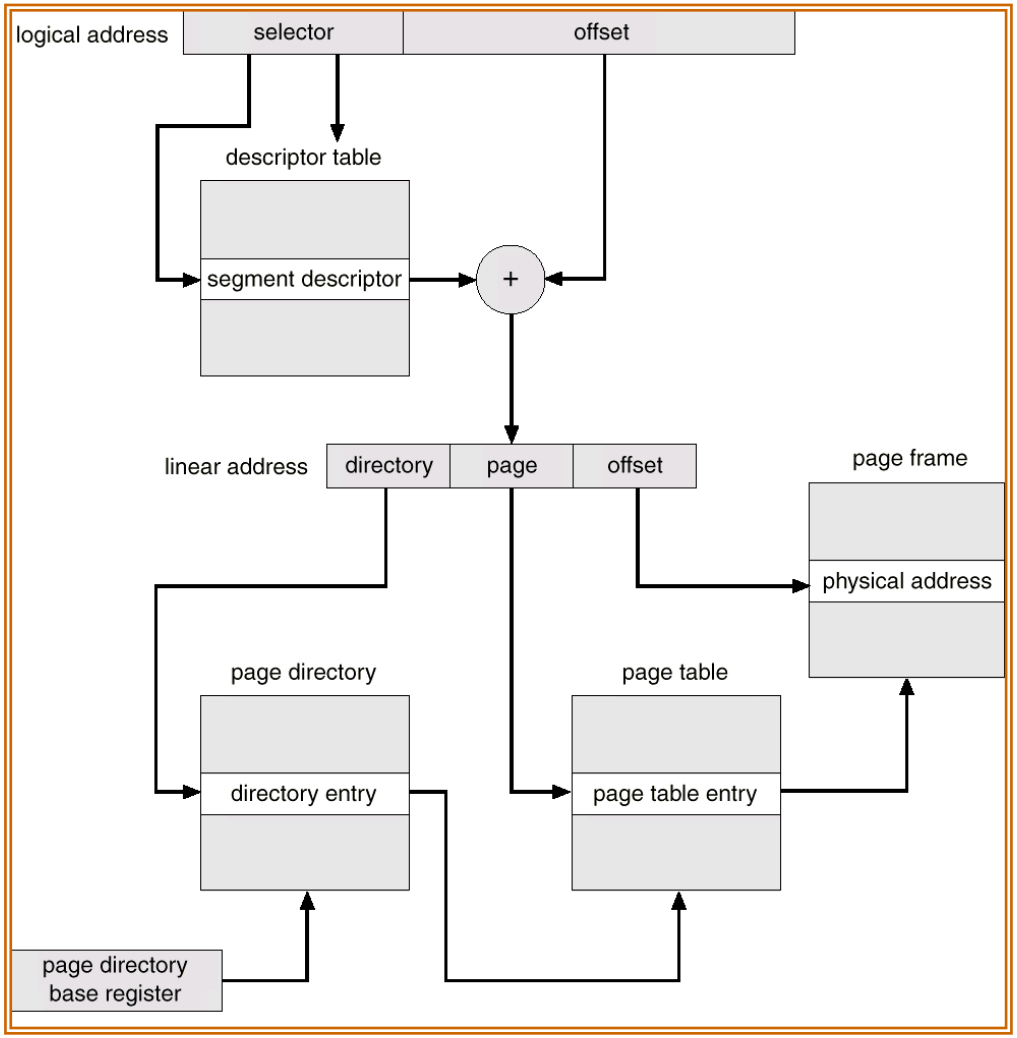


NX No eXecute
AVL AVaiLable to the OS
G Global page
PAT Page Attribute Table
D Dirty (modified)
A Accessed (referenced)

PCD Page Cache Disable
PWT Page Write-Through
U/S User/Supervisor
R/W Read/Write access
P Present (valid)

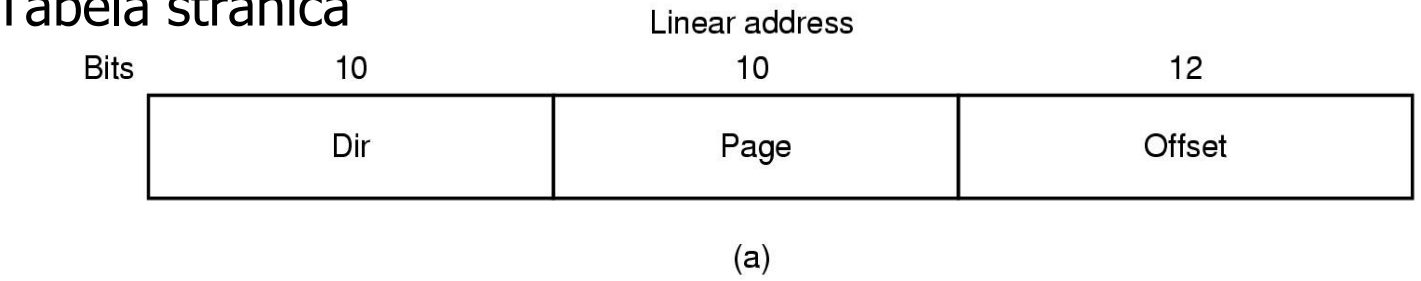
Pentium – Transformisanje adresa (1)

- Virtuelna (logička) adresa
  - Selektor segmenta
  - Ofset u okviru segmenta
- Na osnovu vrednosti selektora pristupa se tabeli deskriptora segmenta
- Na 32-bitnu baznu adresu segmenta dodaje se ofset i formira linearna adresa
- Linearna adresa (32b) se deli
  - Indeks u direktorijum stranica (10 bitova)
  - Indeks u tabelu stranica (10b)
  - Ofset u okviru stranice (12b)

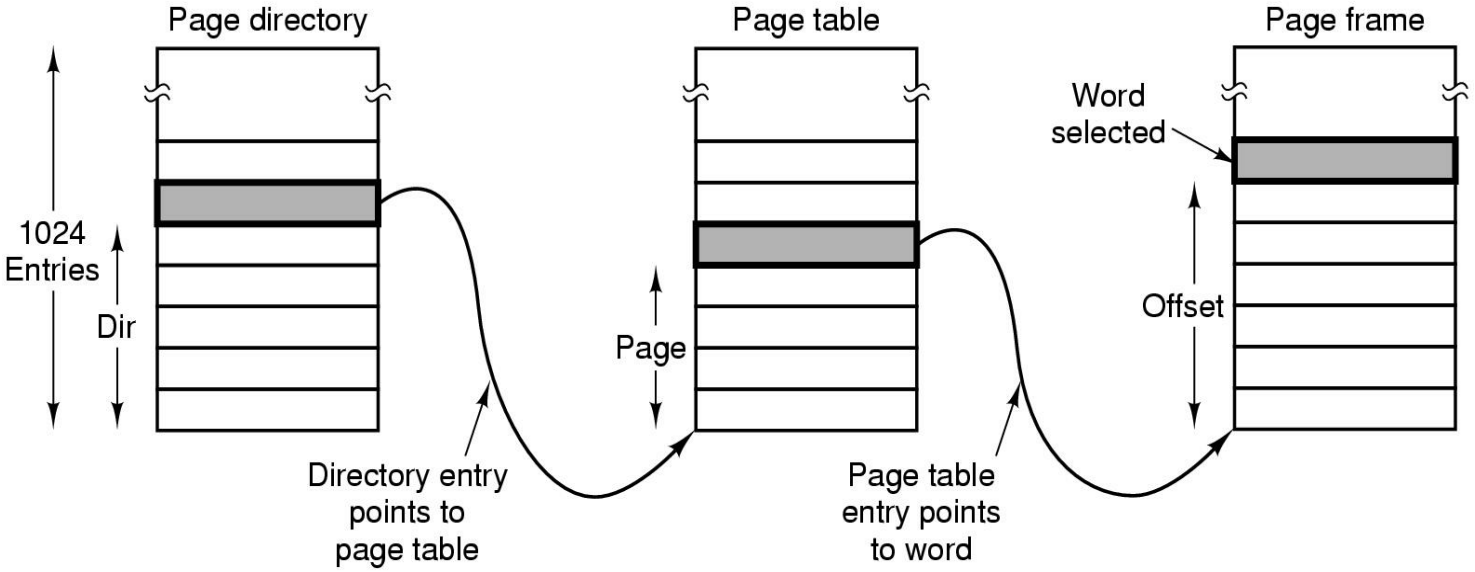


Pentium – Transformisanje adresa (2)

- Tabela stranica u dva nivoa
  - Direktorijum stranica
  - Tabela stranica



(a)



Pentium - Upravljaje memorijom

- Koristi TLB za smeštanje podataka o naskorije korišćenim stranicama u cilju ubrzanja transformacije virtuelne u fizičku adresu (Promašajima se upravlja po algoritmu LRU)
- Pentium MMU arhitektura omogućava da OS implementira
  - Čisto straničenje (Koristi se jedinstveni selektor segmenta, čiji deskriptor ima baznu adresu 0, a veličine segmenta je postavljena na maksimum – 4GB)
  - Čistu segmentaciju
  - Segmentaciju sa straničenjem

Unix & Solaris upravljanje memorijom

- Straničenje – koriste se sledeće strukture podataka
  - Tabela stranica (*Page Table*): Jedna tabela stranica za svaki proces, sa jednom stavkom (ulazom) za svaku stranicu.
  - Deskriptor bloka diska (*Disk Block Descriptor*): Za svaku stranicu postoji stavka koji opisuje disk kopiju te stranice.
  - Tabela podataka okvira stranica (*Page Frame Data Table*): Opisuje svaki stranični okvir glavne memorije, a indeks joj je broj okvira. Koristi se kod algoritama zamene stranica.
  - Tabela swap-ovanja (*Swap Use Table*): Postoji jedna tabela swap-ovanja za svaki uređaj za razmenu (swap), sa jednom stavkom u tabelu za svaku stranicu na uređaju

Unix & Solaris upravljanje memorijom

Page Table Entry

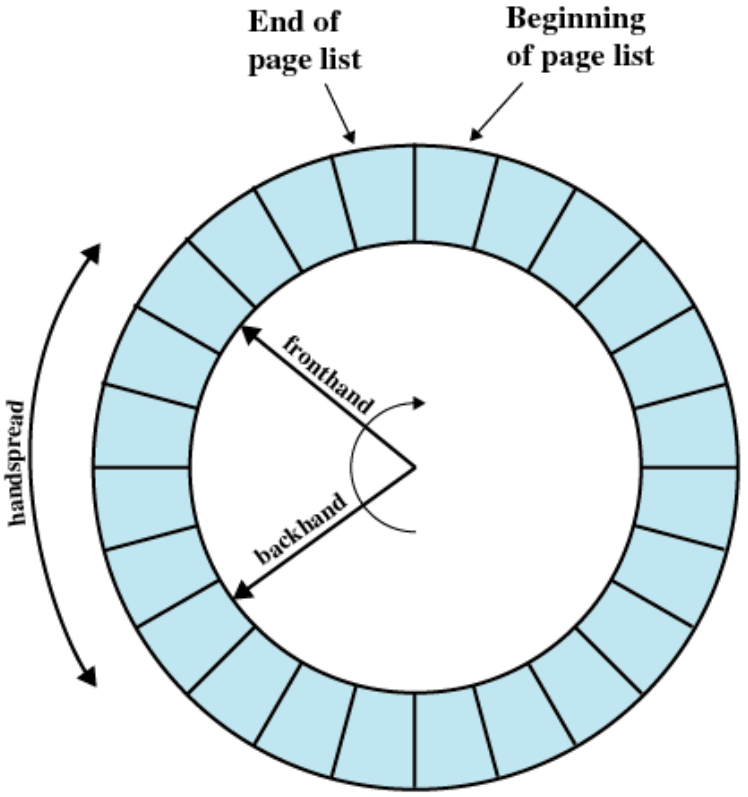
Disk Block Descriptor

Page Frame Data Table Entry

Swap Use Table Entry

Unix & Solaris zamena stranica

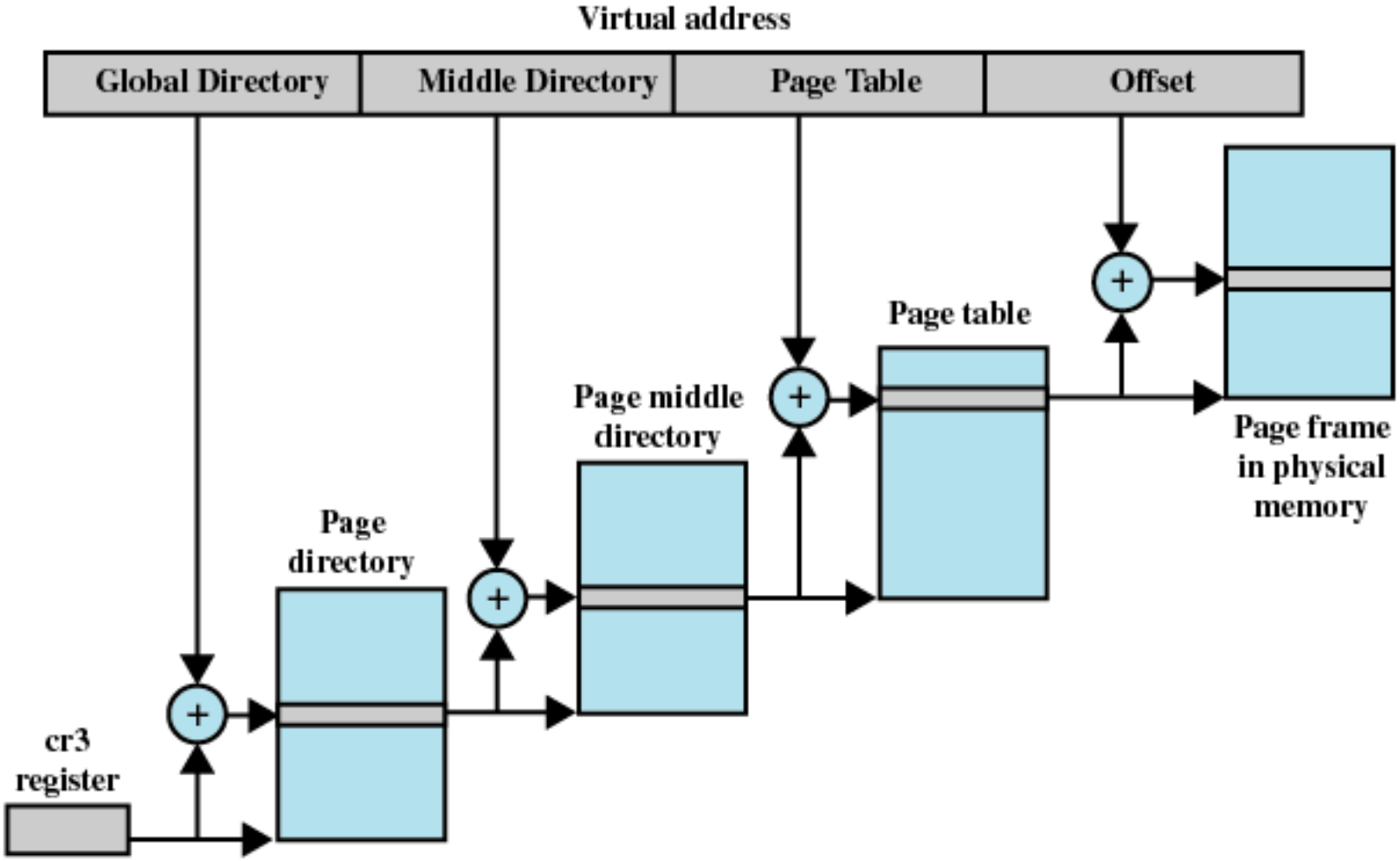
- Tabela podataka okvira stranica koristi se za zamenu
- Svi raspoloživi okviri se povezuju zajedno u listu slobodnih okvira
- Kada broj raspoloživih okvira padne ispod izvesnog praga, oslobađaju se dodatni okviri
- Algoritam časovnika sa dve kazaljke
  - Prednja kazaljka (pokazivač) prolazi preko liste slobodnih stranica i postavlja bit referenciranja (upotrebe) na 0
  - Nešto kasnije druga kazaljka prelazi preko iste liste i proverava bit reference, ako je jednak 1, stranica je skoro referencirana i nije kandidat za zamenu, a ako je jednak 0 ta stranica se smešta na listu slobodnih stranica/okvira



Linux upravljanje memorijom

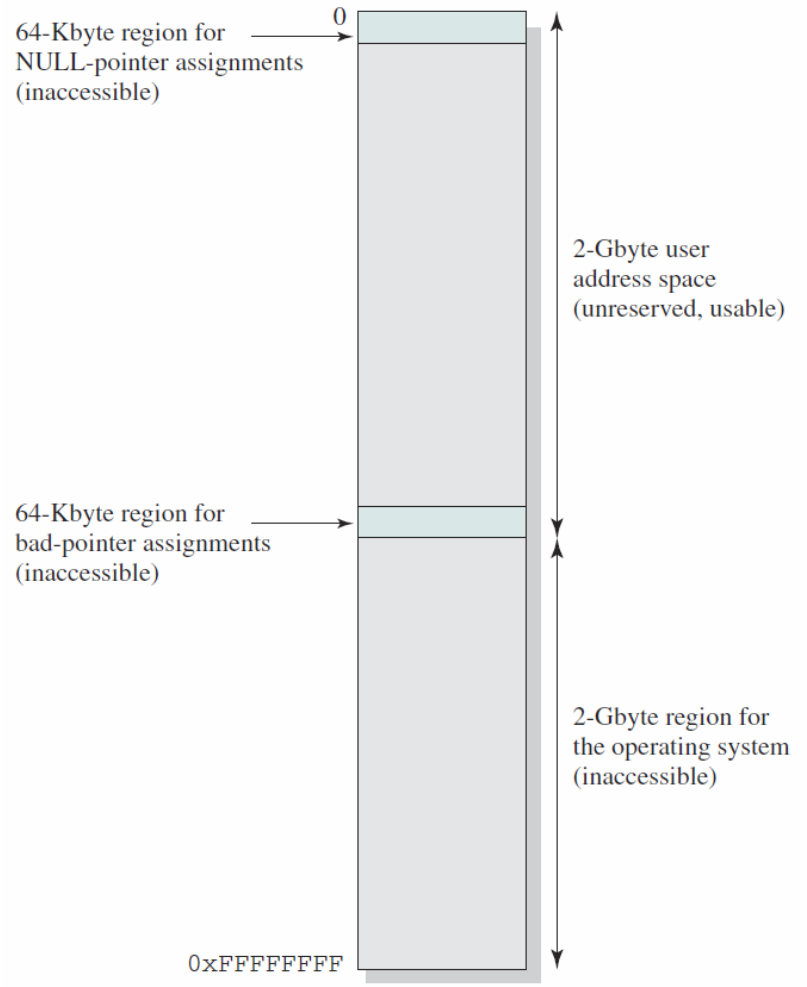
- Straničenje sa tabelama stranica u tri nivoa:
  - direktorijum stranica, srednji direktorijum stranica i tabela stranica
- Dodeljivanje stranica – održava blokove susednih stranica koji su smešteni u susedne stranične okvire (1, 2, 4, 8, 16 ili 32 okvira) – koristi se partnerski sistem (*Buddy system*)
- Algoritam za zamenu po principu časovnika
  - Bit upotrebe je zamenjen 8-bitnim poljem za starost: svaki put kad se pristupi stranici polje za starost se uvećava
  - Linux periodično prolazi kroz globalni skup stranica i smanjuje polje za starost
  - Što je veća vrednost polja za starost stranica je češće referencirana u bliskoj prošlosti; stranica sa starošću "0" je kandidat za zamenu

Linux upravljanje memorijom



Windows upravljanje memorijom

- Procesi rade sa 32 bitnim adresnim prostorom.
- Svaki proces po kreiranju na raspolaganju ima 4GB (2^{32}) virtuelnog adresnog prostora.
- Niža 2GB su na raspolaganju procesu za njegov kod i podatke dok se viša 2GB mapiraju za potrebe jezgra.
- Niža 2GB su privatna za proces i njima ne može pristupiti nijedan drugi proces. Viša 2GB (osim jednog malog dela) su zajednička za sve procese, ali njima ne može direktno da pristupi nijedan proces.



Windows upravljanje memorijom (2)

- Virtual Memory Manager (VMM) – modul koji održava memory map tabelu, upravlja straničenjem i vrši prevođenje virtuelnih u fizičke adrese.
- Stranice su veličine 4KB (Pentium procesori) odnosno 8 ili 16KB (Itanium procesori), a moguća je veličina do 64KB. Ekskluzivno kernel ima pravo da koristi i veće stranice (4MB) kako bi smanjio veličinu memory map tabele. Stranice se učitavaju tek kada se javi zahtev za njima (*demmand paging*)
- Koristi se straničenje na zahtev u kombinaciji sa tehnikom *clustering*. Kada proces referencira neku stranicu, postoji velika verovatnoća (zbog sekvencijalne prirode programa i podataka) da će u bliskoj budućnosti referencirati susednu stranicu. Kada se desi *page fault* VMM radi učitavanje (*load*) referencirane stranice i nekoliko susednih stranica.
- Za zamenu stranica koristi se algoritam radnog skupa (working set), dok se radnim skupom upravlja tehnikom ***promenljivo dodeljivanje, lokalni opseg***.

Domaći zadatak

- Poglavlje **8 Virtuelna memorija**
  - 8.8 Ključni pojmovi, kontrolna pitanja i problemi
- Paging & segmentation animations
  - https://apps.uttyler.edu/Rainwater/COSC3355/Animations