

Operacijski sistemi

Virtualizacija
pomnilnika

Vsebina

- Virtualizacija pomnilnika
- Ostranjevanje
- Uporabna zunanjega pomnilnika

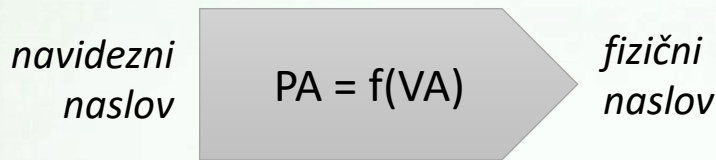
Virtualizacija pomnilnika

- Cilji

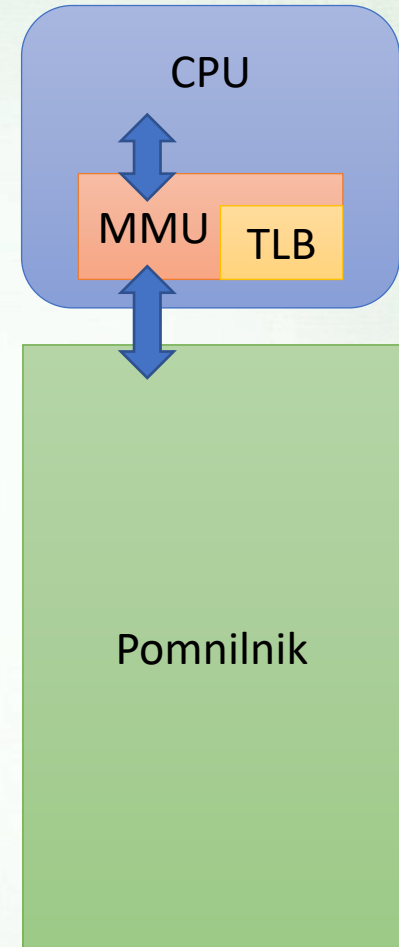
- proces vidi sklenjen kos pomnilnika
 - lahko gre za zelo velik kos
- izolacija procesov
 - procesi ne morejo vplivati drug na drugega
 - tudi OS je izoliran od procesov
 - uporaba mehanizma zaščite (strojna podpora)
- transparentnost
 - proces se ne zaveda virtualizacije
- učinkovitost
 - preslikava naslovov mora biti hitra (strojna podpora)

Virtualizacija pomnilnika

- Preslikava naslovov



- podpora strojne opreme
 - OS skrbi za konfiguracijo
 - MMU izvaja preslikovanje
 - TLB skrbi za učinkovitost preslikovanja



Virtualizacija pomnilnika

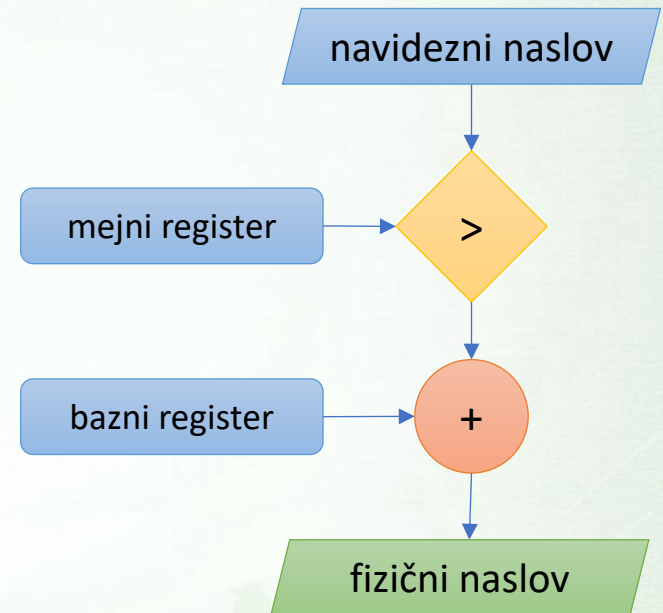
- Izvedbe preslikovanja
 - **celotna preslikovalna tabela**
 - izvedba za vse navidezne naslove je nemogoča
 - **premeščanje** (relocation)
 - naslovni prostor premestimo na poljubno mesto v fizičnem pomnilniku
 - **segmentacija** (segmentation)
 - razdelitev pomnilnika na bloke različnih velikosti
 - **ostranjevanje** (paging)
 - razdelitev pomnilnika na bloke enake velikosti
 - **segmentacija in ostranjevanje**
 - segmenti so razdeljeni na strani

Virtualizacija pomnilnika

- **Statično premeščanje**
 - program se prevede na naslov 0
 - **nalaganje** programa
 - program se naloži na izbrani **ciljni fizični naslov**
 - ponoven preračun in prepis naslovnih operandov
 - **težavna zaščita**
 - **statični naslovi:** ok, **dinamični naslovi:** težava
 - **težavno ponovno premeščanje** programov
 - uporaba, le če ni strojne podpore za virtualizacijo pomnilnika

Virtualizacija pomnilnika

- **Dinamično premeščanje**
 - nalaganje programa
 - program se naloži na izbrani ciljni fizični naslov
 - **bazni register** hrani nalagalni naslov
 - **mejni register** hrani velikost naslovnega prostora
 - enostavno preslikovanje
 - enostavna zaščita
 - enostavno premeščanje



Virtualizacija pomnilnika

- **Dinamično premeščanje** – strojna podpora
 - preslikovalna enota
 - bazni (base) in mejni (bounds/limit) register
 - izvedba preslikovanja in preverjanja zaščite
 - pasti
 - prekoračitev mejnega registra: OS ukine proces
 - izvedba privilegiranih ukazov: OS ukine proces
 - posebni strojni ukazi
 - uporaba le s strani OS (privilegirani način)
 - manipuliranje baznega in mejnega registra
 - namestitev pasti

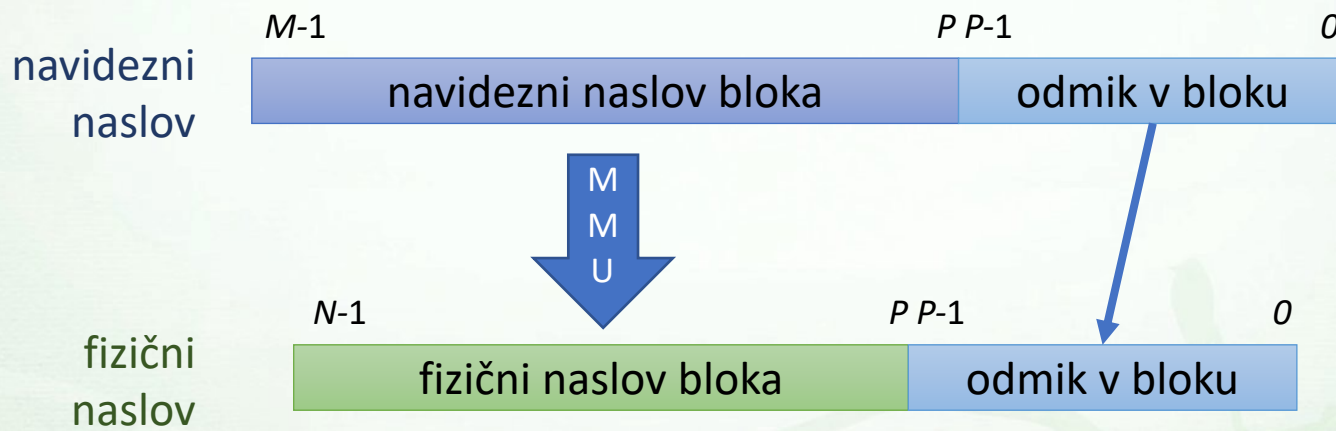
Virtualizacija pomnilnika

- **Segmentacija in odstranjevanje**

- pomnilnik razdeljen na *bloke* velikosti 2^P

- **preslikovanje**

- iz M bitnega navideznega naslovnega prostora
- v N bitni fizični naslovni prostor




Virtualizacija pomnilnika

• Segmentacija

- bloki: segmenti
- segmenti so različne velikosti
- segmenti imajo nek pomen
 - npr. koda, podatki itd.
- težavno upravljanje pomnilnika
 - iskanje prostora za alokacijo segmenta
- zunanja fragmentacija

• Ostranjevanje

- bloki: strani in okvirji
- strani so enake velikosti
- strani nimajo posebnega pomena
- lažje upravljanje pomnilnika
 - enostavno iskanje prostega okvirja strani
- notranja fragmentacija



Veliki OS običajno uporabljajo ostranjevanje.

Ostranjevanje

- **Stran**

- blok v navideznem naslovnem prostoru

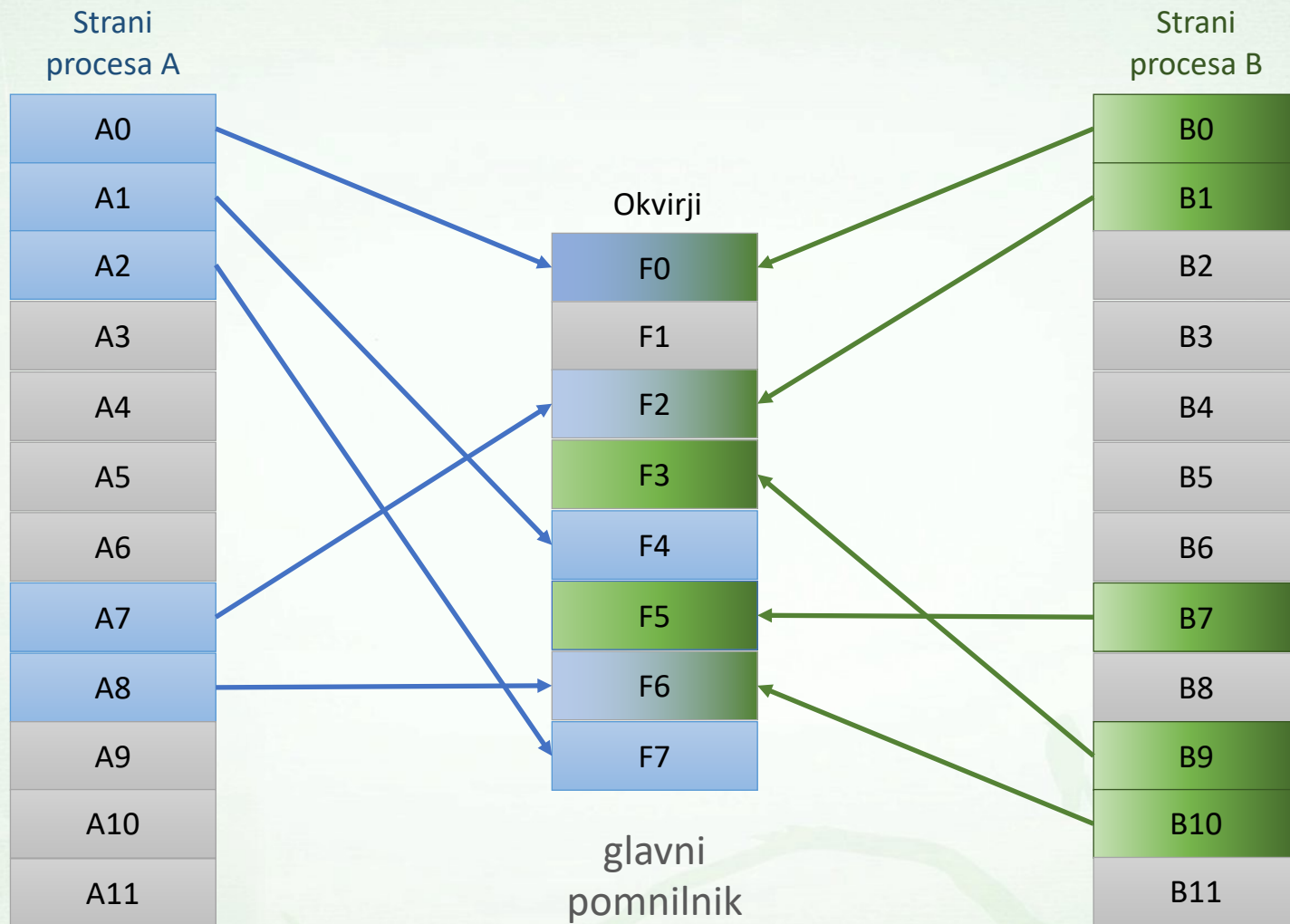
- **Okvir strani**

- blok v fizičnem naslovnem prostoru
 - okvir hrani poljubno stran
 - okvirji so enake velikosti kot strani

- **Preslikovanje**

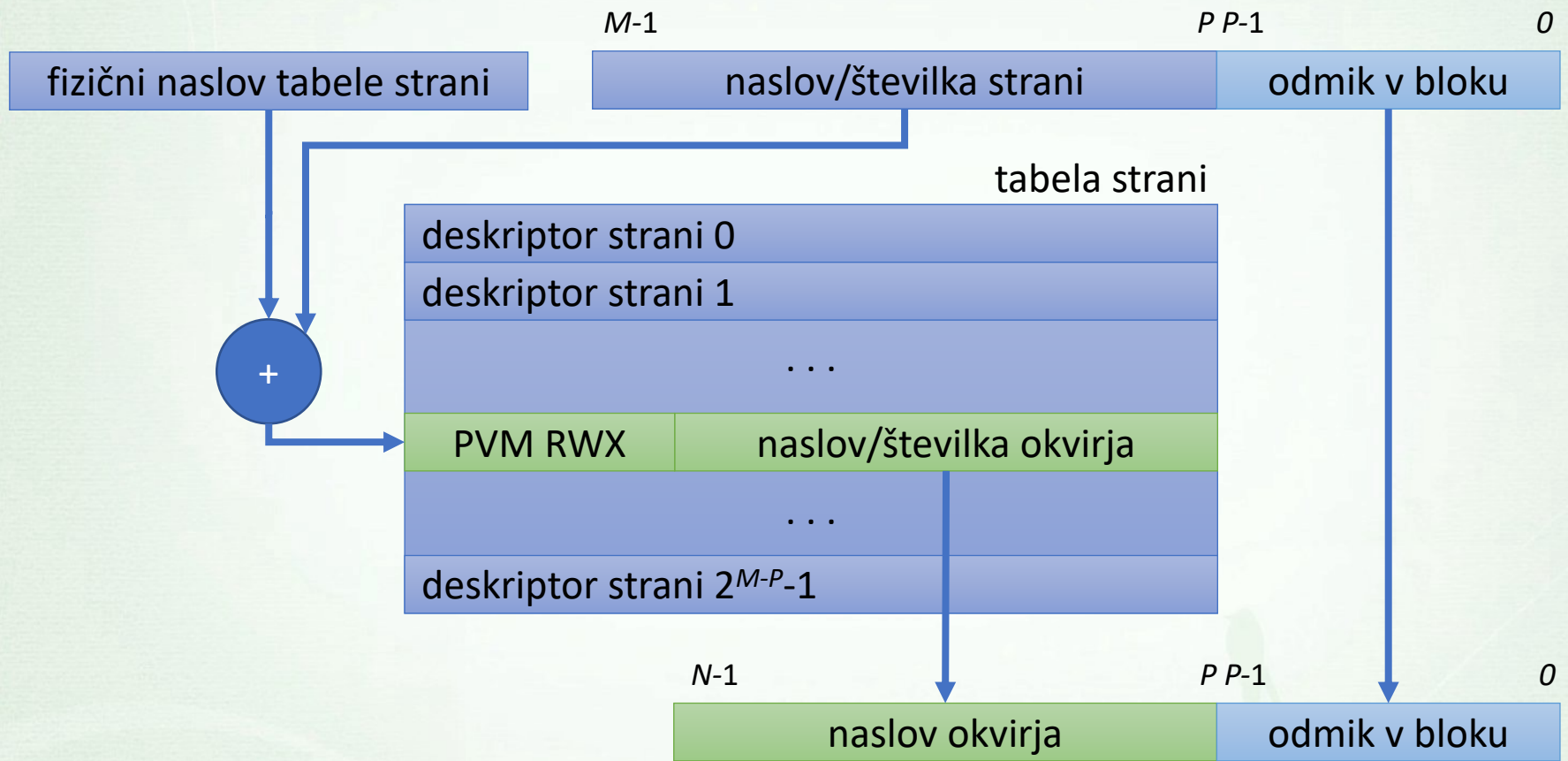
- navidezni naslov (naslov strani + odmik)
preslikamo v
fizični naslov (naslov okvirja + odmik)

Ostranjevanje



Ostranjevanje

- Preslikovanje s tabelo strani



Ostranjevanje

- Velikost tabele strani
 - 2^{M-P} deskriptorjev v tabeli

P	2^P	$2^{M-P}, M = 32$	$2^{M-P}, M = 64$
12	4 KiB	$2^{20} = 1048576$	2^{52}
22	4 MiB	$2^{10} = 1024$	2^{42}

Primer iz x86/Linux:

- $P = 12$: 4 KiB strani
 - $M = 32$: 32 bitni naslovni prostor
 - npr. 4 bajtni deskriptor
 - 4 MiB za tabelo strani na proces
- rešitvi
 - večnivojska tabela strani
 - invertirana tabela strani

Ostranjevanje

- Učinkovitost preslikovanja
 - ena preslikava
 - št. dostopov do pomnilnika
= št. dostopov do tabel in podtabel strani + 1
 - rešitev
 - TLB – translation lookaside buffer
 - preslikovalnik predpomnilnik (SRAM v MMU enoti)
 - **prostorska lokalnost**
 - podatki so blizu skupaj v pomnilniku (na isti strani)
 - **časovna lokalnost**
 - ponoven dostop istih podatkov

Ostranjevanje

- Naloge OS
 - vodenje evidence prostih okvirjev strani
 - upravljanje preslikovalnih tabel
 - stvaritev procesa: inicializacija naslovnega prostora
 - končanje procesa: sprostitvev vseh zasedenih okvirjev
 - preslikava pomnilnika (mmap): izvedba dodatne preslikave
 - praznenje TLB ob preklopu procesa

Uporaba zunanjega pomnilnika

- Pomnilniška hierarhija
 - notranji oz. fizični pomnilnik: RAM
 - zunanji pomnilnik: trdi disk ali SSD
- Izziv
 - Čeprav je na voljo le *malo* notranjega pomnilnika, je naslovni prostor procesov zelo velik.

Kako procesu zagotoviti
navidezni naslovni prostor večji
od velikosti fizičnega pomnilnika?

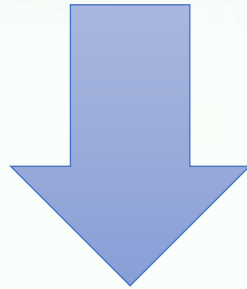


Uporaba zunanjega pomnilnika

- **Pomnilniški prekrivki (memory overlays)**
 - star programski pristop
 - celoten program je prevelik za pomnilnik
 - program razdeljen na več ustrezno velikih kosov
 - posamezni kosi so shranjeni v datotekah
 - programer sam poskrbi za sproščanje pomnilnika in nalaganje nove kode, ki prekrije staro

Uporaba zunanjega pomnilnika

- Večprogramiranje
 - **več procesov** hkrati naloženih v *premajhnem* pomnilniku: uporaba zunanjega pomnilnika je nujna

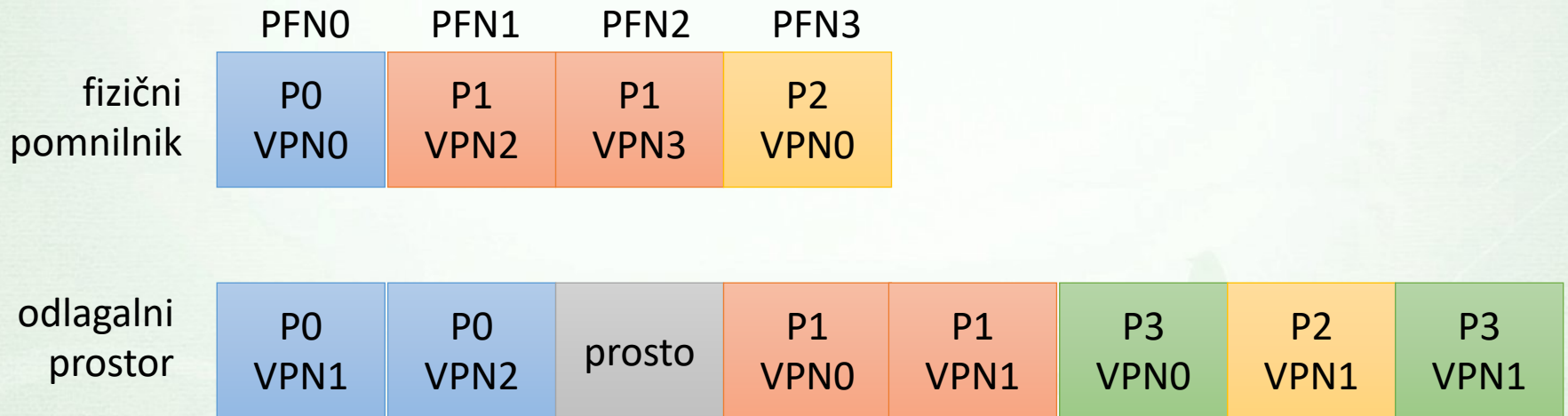


- **Odlagalni prostor (swap space)**
 - del zunanjega pomnilnika
 - swap out
 - shranjevanje strani iz pomnilnika na disk
 - swap in
 - nalaganje strani iz diska v pomnilnik

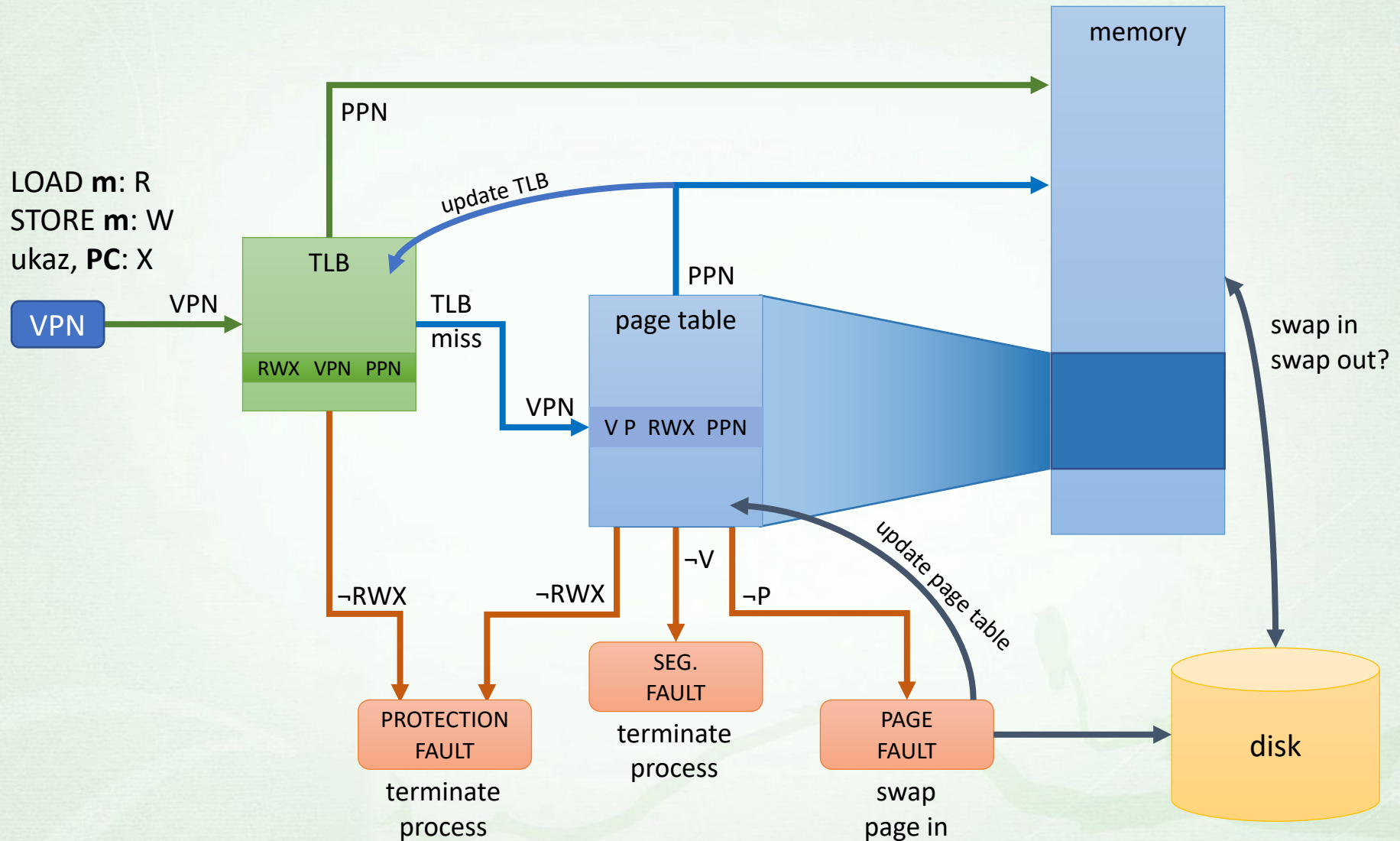
Uporaba zunanjega pomnilnika

- Primer – 4 procesi

- P0: 3 strani: 1+2
- P1: 4 strani: 2+2
- P2: 2 strani: 1+1
- P3: 2 strani: 0+2



Uporaba zunanjega pomnilnika



Uporaba zunanjega pomnilnika

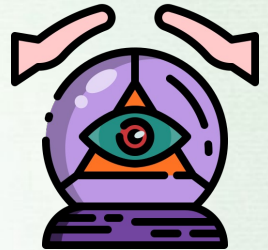
- Postopek naslavljanja podatka
 - zadetek v TLB
 - preveri, če VPN obstaja v TLB
 - če obstaja, potem naslovi ustrezni del fizičnega pomnilnika
 - zgrešitev v TLB, iskanje v tabeli strani
 - preveri, če VPN obstaja v preslikovalni tabeli strani
 - če je zapis veljaven (valid) in stran obstaja (present), potem naslovi ustrezni del fizičnega pomnilnika
 - zgrešitev strani (page fault, page miss)
 - naloži stran iz zunanjega pomnilnika
 - osveži vnos v preslikovalni tabeli strani
 - ponovi strojni ukaz

Uporaba zunanjega pomnilnika

- **Kaj** če je pomnilnik v celoti zaseden?
 - zgrešitev strani, vendar ni prostora za novo stran
 - potem pride do zamenjave strani
 - izbrana stran gre ven (swap out)
 - nova stran gre noter (swap in)
 - izbira strani (page replacement policy)
 - minimizacija števila zgrešitev strani
 - page (out) daemon
 - high / low watermark

Uporaba zunanjega pomnilnika

- Algoritmi izbire strani za izločitev
 - **FIF** – furthest in future, optimalni algoritem
 - izločitev strani, ki bo dostopana najdlje v prihodnosti
 - zgrešitev strani prestavimo karseda v prihodnost
 - zahteva poznavanje prihodnosti
 - **FIFO** – first in, first out
 - izločitev strani, ki je najdlje v pomnilniku
 - izziv: stare strani se lahko pogosto uporabljajo
 - **Naključna zamenjava**
 - izločitev naključne strani



Uporaba zunanjega pomnilnika

- Algoritmi izbire strani za izločitev
 - **LFU** – least-frequently used
 - izloči najmanj uporabljano stran
 - potrebno evidentirati frekvenco uporabe
 - **LRU** – least-recently used
 - izloči najdlje neuporabljeno stran
 - **izvedba: časovni zaznamek**
 - za vsak vnos v preslikovalni tabeli strani
 - osvežitev zaznamka pri vsakem dostopu do strani
 - **izvedba: vrsta strani**
 - dostopano stran pomaknemo na začetek vrste
 - ob vsakem dostopu do strani

Uporaba zunanjega pomnilnika

- Strojna pomoč
 - dodatni biti v vnosu tabele strani
 - **modified/dirty bit**
 - če je bila stran spremenjena od zadnjega nalaganja
 - nespremenjenih strani ni potrebno shraniti nazaj na disk
 - **fixed bit**
 - preslikava je fiksna, pusti pri miru
 - za dostop do V/I preslikanih naprav
 - **access/reference**
 - ali je bila stran dostopana (branje ali pisanje)
 - osnova za gradnjo približnih LRU algoritmov