ŘÍZENÍ OPERAČNÍ PAMĚTI

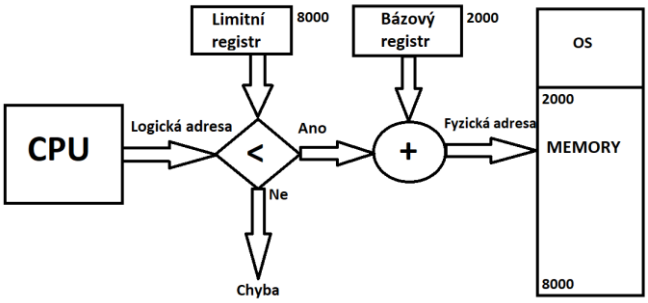
- **K čemu slouží operační paměť**
- **Co se nachází v operační paměti**
- **Strategie přidělování operační paměti**
  - **Souvislá oblast**
  - **Po blocích**
  - **Stránkování**
  - **Segmentace**
  - **Segmentace se stránkováním**
- **Výpadek stránky, pre-cleaning, thrashing**
- **Čisté a špinavé stránky**
- **Algoritmy výměny stránek**
  - **Optimální algoritmus**
  - **FIFO**
  - **LRU**
  - **Druhá šance**
  - **Hodiny**
  - **NRU**
  - **Random**
  - **NFU**

K čemu slouží operační paměť a co se v ní nachází?

- Běžně označována zkratkou RAM = Random Access Memory, což znamená paměť s nahodilým přístupem dat
- Patří do kategorie RWM paměti = Read Write Memory, to jsou paměti, ze kterých lze číst i do nich zapisovat data a jsou závislá na napájení, takže po odpojení z elektrické sítě ztrácí data
- Slouží jako dočasné úložiště dat
- Obsahuje běžící programy a aktuálně zpracovávaná data
- Je mnohem rychlejší než vnější paměť (HDD)

Strategie přidělování místa v paměti – přidělení veškeré volné paměti (souvislá oblast)

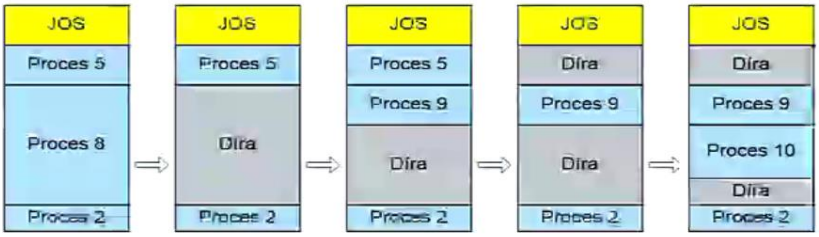
- Operační paměť rozdělená na dvě části
  - Část operační paměti je obsazena operačním systémem (kód programu, vyrovnávací paměť,…)
  - Zbytek je k dispozici pro uživatelský program
- V každém okamžiku je v paměti nejvýše jeden uživatelský program
- V bázovém registru si určím počáteční adresu
- Veškeré logické adresy, které přicházejí, se mi k nim přičte hodnota z bázového registru
- Tento způsob přidělování umožnuje ochranu, tzn., pokud bych překročil nějaký limit (ten je nastaven v limitním registru), nebude přidělení možné
- Nevýhodou je, že není využita celá paměť, výhodou je jednoduchost
- Příklad: mám v paměti místo od počáteční adresy 2000 až do 8000. Hodnota 2000 je uložena v bázovém registru, hodnota 8000 v limitním registru. Když budu adresovat buňku 50, k hodnotě 50 se mi přičte hodnota z bázového registru 2000, tudíž fyzická adresa bude 2050. Před sečtením ale proběhne testování, v případě, že by buňka byla větší než hodnota v limitním registru, adresování do paměti by neproběhlo.



- V případě, že se mi všechny procesy nevejdou do operační paměti, využije se swapování, kdy je přidělen veškerý uživatelský prostor (od konce OS do konce OP) jednomu procesu a všechny ostatní procesy jsou na úložiště (HDD). Pokud chci zavolat druhý proces (z uložiště) nejprve se odloží stávající proces z OP do uložiště a je nahrazen tím druhým
- Další metoda je prokládání, tu použito v případě, že proces má větší velikost, než je velikost samotné OS paměti. Daný proces je rozdělen na dvě části – část, která musí být trvale v paměti, a časti, které mohou být překrývány, tak aby součet vešel do paměti. Překrývání musí být dáno uživatelem, nikoliv určen OS.

Strategie přidělování místa v paměti – po blocích

- Kromě OS se v paměti může nacházet více než 1 proces
- Paměť je rozdělena na bloky (sekce) o stejné velikosti
- Nutno je dopředu znát velikost úlohy
- Správce paměti je schopen spojit volné bloky (sekce), ale pouze ty, které na sebe navazují -> umožní uložit větší/náročnější proces
- Typickým příkladem je MS DOS
- Výhoda je jednoduchá implementace
- Nevýhoda je značná fragmentace a nutnost změny stavových registrů (není možné, aby OS kontroloval všechny registry při každém zápisu do paměti)
- Fragmentaci lze odstranit tak, aby volné paměti ležely vedle sebe – tomuto se říká setřásání, kdy správce paměti přesune paměťové bloky za sebou
- Statické a dynamické přidělování
    - Statické přidělování sekce
    - Dynamické přidělování sekce
        - Sekce jsou vytvářeny až při vzniku úlohy (procesu)
        - Řeší problém defragmentace
        - Periodicky slučuje volné oblasti do jedné
          Výhodou je eliminace fragmentace a možnost více sekcí

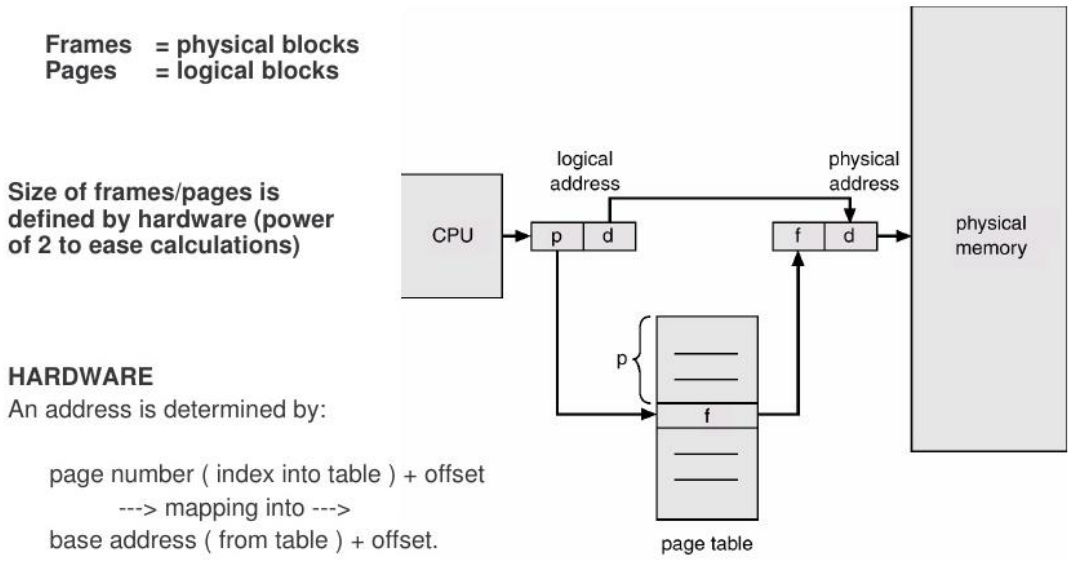


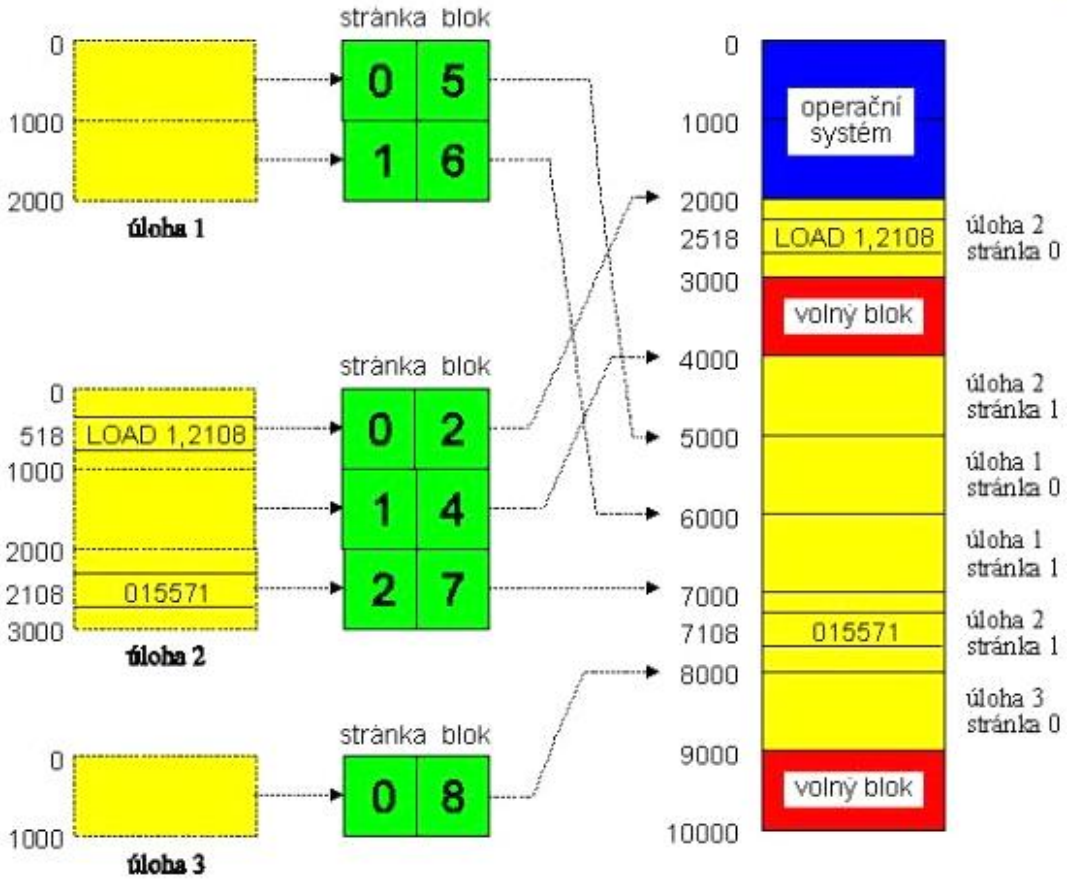
        - Po konci přidělování paměti vzniknou díry po již ukončených procesech, pokud by přišel proces, který by se vešel do paměti, kdyby se obě díry (místa) sečetly, musím provést defragmentaci (setřepání) prostoru

- Alokační strategie
    - FIRST FIT
        - Obsadí 1. volný blok, do kterého se proces vejde
        - Nejčastější a nejjednodušší na implementaci
    - BEST FIT
        - Obsadí nejvhodnější blok -> zůstane málo volného prostoru
    - LAST FIT
        - Obsadí poslední volný blok
    - WORST FIT
        - Neřeší nic, umístí se do největšího vyhovujícího volného místa

Strategie přidělování místa v paměti – stránkování

- Logická paměť je rozdělena na rámce o stejné velikosti
- Proces je rozdělen na stránky o stejné velikosti
- Číslování stránek začíná vždy od 0
- Poslední stránka procesu bývá nevyužitá celá
- Každá stránka má registr (tabulka stránek)
- Logická adresa se skládá z offsetu stránky a čísla stránky
    - Offset stránky je relativní adresa
    - Číslo stránky je index do tabulky stránek obsahující bázovou adresu rámce
- Při výpočtu fyzické adresy se číslo stránky odkáže na tabulku stránek, vybere se příslušná stránka, ve které je uložena skutečná fyzická adresa, ta se sečte s offsetem a odkáže do fyzické paměti
- Pro velikost stránky 4 KB je offset potřeba 12 bitový (2^{12}= 4K)
- Pro příslušný proces se vyhledá rámec
- Ukládání stránek do rámců probíhá náhodně
- Procesy pro svůj běh potřebují souvislý úsek paměti
    - Stránkování umožnuje přidělit procesu několik nesouvislých úseků a vytvořit iluzi, že se jedná o společný prostor
- Při jejich uvolňování dochází k fragmentaci
    - Řešením je dynamické přemísťování bloků
- Do 1 rámce se uloží 1 stránka
- Výhody
    - Odstranění fragmentace
    - Není nutné přemísťování bloků paměti
- Nevýhody
    - Poslední stránka procesu nebývá využitá celá, a proto velikost stránky nesmí být příliš velká -> vnitřní fragmentace
    - Nutná HW podpora

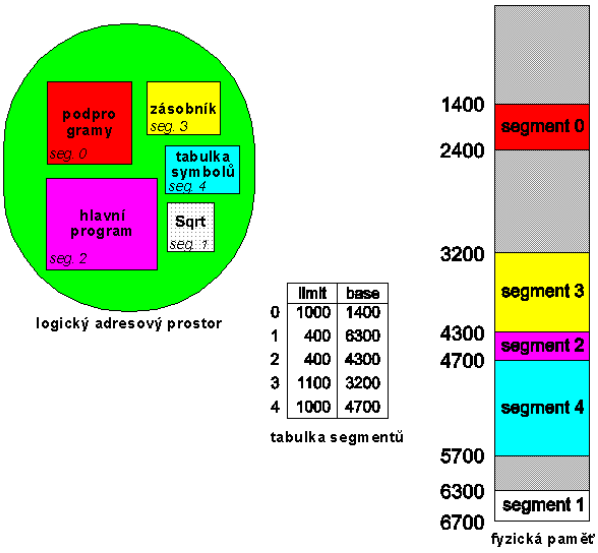




- **Stránkování na žádost**
  - Z důvodu uvolnění fyzické paměti a umožnění tak využití pro další procesy jsou některé stránky odloženy na disk do tzv. swapovacího oddílu
  - V tabulce stránek je umístěn údaj, zda je stránka na disku nebo OP
  - Pokud je stránka na disku, dojde k vyvolání výpadku stránky -> obslužný program pro toto přerušení musí zajistit nahrání stránky do OP, opravit údaj v tabulce stránek a zajistit opakované volání instrukce
  - V případě obsazenosti paměti je nutné odložit jiný rámec na disk -> algoritmy nahrazování stránek

Strategie přidělování místa v paměti – segmentace

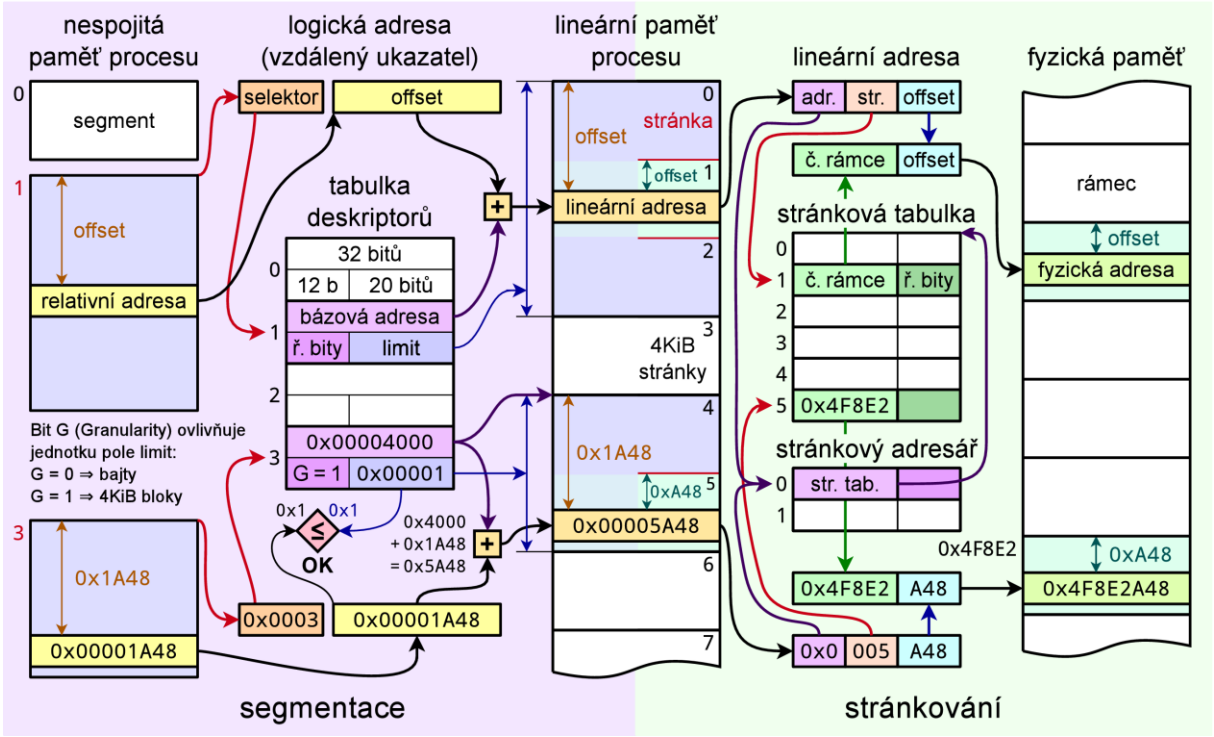
- Program se skládá z několika segmentů
- Segment se může lišit svou velikostí
- Segment je logické seskupení informací (hlavní program, zásobník, konstanty,…)
- Číslování segmentů je od 0, procesu se tak jeví jako jeden logický souvislý prostor
- Logická adresa (odkaz na paměť) se skládá z čísla segmentu a offsetu
  - Offset nesmí být větší, než je velikost segmentu
- V segmentové tabulce najdeme bázovou adresu a limit
  - Bázová adresa = base nám říká, kde začíná v OP
  - Limit nám říká, jak je segment velký
- Tabulka segmentů je uložená v operační paměti
- Procesor vysílá požadavek na logickou adresu
- Pokud se dohledá v tabulce segmentů, číslo segmentů se změní na bázovou adresu
- Segmentace umožní, aby procesy, které jsou řízený stejným programem, sdílely kód programu, ušetří tak místo v paměti
- SHRNUTÍ
  - Daný proces rozdělím do jednotlivých segmentů o různé velikosti a umisťuju je do paměti na základě tabulky segmentu. V tabulce segmentu se zapisuje začátek segmentu = base a délka segmentu = limit. Vidím, že segment 0 začíná na adrese 1400 a má délku 1000. Mám-li v segmentu 0 adresu např. 400, 0 mi ukáže, kde se nachází base, vezmu base 1400 a přičtu k adrese 400, tedy výsledná adresa bude 1800, tedy na adrese 1800 se mi bude nacházet příslušná adresa buňky



- Výhody
  - Velikost segmentů odpovídá skutečné potřebě části procesu (potlačena vnitřní fragmentace)
  - Možnost detekce chyb v programech na základě offsetu, který je porovnáván s limitem
  - Sdílení kódu mezi programy
- Nevýhody
  - Náročnost alokace segmentů (různé délky, víc programů současně)
  - Nutná HW podpora
  - Při výpadku nutno vyměnit celý segment

Strategie přidělování místa v paměti – segmentace se stránkováním

- Proces je rozdělen na segmenty
  - Segment je hlavní program, podpogram, konstanty, ….
- Segmenty jsou číslovány od 0
- Segmenty jsou dále rozděleny na stránky
- Stránky jsou uloženy v paměti v podobě rámců
- Velikost stránky i rámce je 4 KB
- Procesor pošle požadavek na logickou adresu, která je 48 bitová
  - Skládá se z offsetu 32 bit a selektoru 16 bit – z toho 13 bit je index a další 3 bity jsou tabulkový indikátor a úrovně oprávnění
- Selektor vybírá index, ten ukazuje do segmentové tabulky
- Pro konkrétní segment tam najdeme limit 20 bit, bázovou adresu 32 bit a řídící bity 12 bit
- Limit je porovnáván s offsetem, jestli náhodou nepřesahuje maximální velikost
- Bázová adresa a offset jdou do součtového členu -> 32 bit + 32 bit = 32 bitová lineární adresa, která je rozdělena na 10 bit stránkovací tabulku, 10 bitovou stránkovací adresu a 12 bitový offset
- Použito dvojúrovňové stránkování
- Číslo stránky dohledá ve stránkovací tabulce číslo rámce, který se spojí s offsetem a vznikne fyzická adresa
- Jsou použity 2 offsety!!!!
  - Jeden je posun v rámci segmentu
  - Druhý je posun v rámci stránky (to je ten menší 12 bitový)



Čisté a špinavé stránky

- Čisté stránky není nutno kopírovat na disk, neboť jsou shodné
- Špinavé stránky jsou modifikované stránky a jsou kopírovany do paměti

Čištění – Precleaning

- V případě, že má systém čas, zálohuje si data do virtuální paměti
- Pokud dojde k modifikaci, nutno zálohovat znova

Výprask - Thrashing

- Problém, kdy by OS přehazoval stránky mezi virtuální a operační paměti
- Řešením je zvýšit kapacitu paměti

Krátký výprask – Swap Storm

- Může nastat při nedostatku operační paměti
- Řešením je změna algoritmu, snížení počtu aplikací

Operační systémy

- WIN7 a XP stránkování na žádost
- V Linuxu segmentace nebo stránkováni na žádost s využitím algoritmu NRU
- Mac OS využívá stránkování na žádost

Algoritmy

- Optimální algoritmus
  - Nahrazení stránky, která bude volána ze všech nejpozději
  - Není možné jej naprogramovat, z důvodu, že nevíme, jaké budou požadavky do budoucna
  - Pokud bude paměť vyčerpána, swapuje se stránka, která bude volána nejpozději
- FIFO
  - First in first out
  - První stránka, která přišla, první odejde
  - Pokud bude paměť vyčerpána, swapuje se stránka, která je ve frontě nejdéle
- LRU
  - Last Recently Used
  - Nejméně používána půjde pryč (dívá se do minulosti)
  - Pokud bude paměť vyčerpána, swapuje se stránka, která je nedávno nejméně používána
- Druhá šance
  - Vylepšený FIFO algoritmus
  - Obsahuje ručičku, která ukazuje na to, která stránka by mohla jít pryč
  - Stránky jsou řazeny v kruhovém seznamu
  - Kouká se na začátek fronty a kontroluje, jestli není nastaven referenční bit
  - Pokud není, stránka se nahradí, pokud je, jde zpět do fronty
  - Snaha zabránění vyhození často používané stránky
- Hodinový algoritmus
  - Modifikace druhé šance, přibyde ručička, která ukazuje na nějakou stránku
  - Pokud tam bude značka, ručička se posune dál
  - Značku dostane, pokud bude opětovně volána
  - Při zápisu nové stránky se ručička posune o jedno dál
  - Když je stránka opětovně volána a dostává značku, ručička se neposune
- Random
  - Náhodně vyhodí stránku
- NFU
  - Not frequently used
  - Čítač stránek, periodicky navyšován
- NUR
  - Not used recently
  - Nepoužitá stránka půjde pryč
  - Obsahuje dva modifikátory

Konkrétní příklady

Optimální algoritmus (7 výpadků)												
požadavek	1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	1	1	1	1	1	1	3	3	3
2		2	2	2	2	2	2	2	2	2	4	4
3			3	4	4	4	5	5	5	5	5	5

FIFO (9 výpadků)												
požadavek	1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	4	4	4	5	5	5	5	5	5
2		2	2	2	1	1	1	1	1	3	3	3
3			3	3	3	2	2	2	2	2	4	4

LRU (10 výpadků)												
požadavek	1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	4	4	4	5	5	5	3	3	3
2		2	2	2	1	1	1	1	1	1	4	4
3			3	3	3	2	2	2	2	2	2	5

Druhá šance (8 výpadků)												
požadavek	1	2	3	1	4	2	5	1	2	5	1	4
1	1	1	1	1*	1	1	5	5	5	5*	5*	5
2		2	2	2	4	4	4	1	1	1	1*	1
3			3	3	3	2	2	2	2*	2*	2*	4