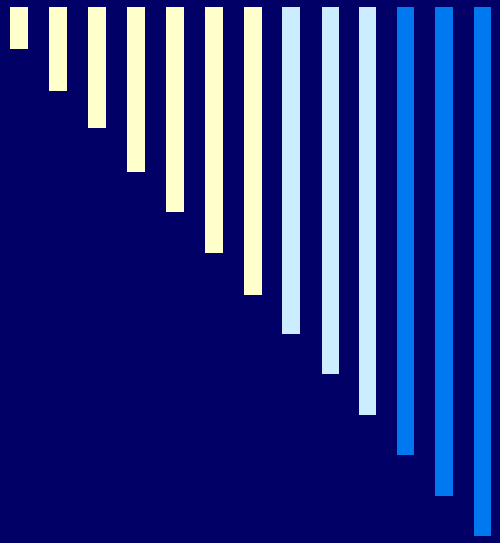
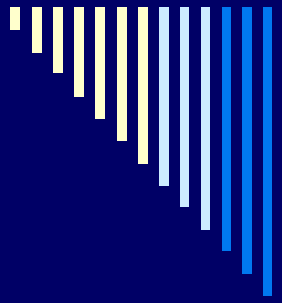


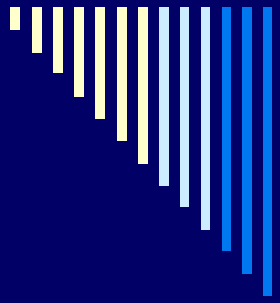
02. Koncepce OS
Procesy, vlákna

ZOS, L. Pešička



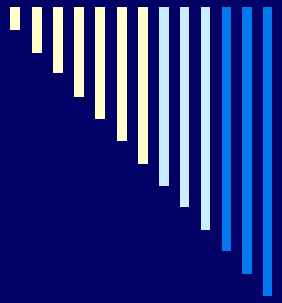
Koncepce OS

- Základní abstrakce
  - procesy
  - soubory
  - uživatelská rozhraní



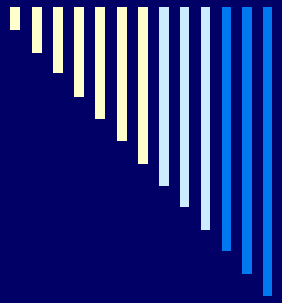
Procesy

- **Proces** – instance běžícího programu
- **Adresní prostor** procesu
  - MMU zajišťuje soukromí
  - kód spustitelného programu, data, zásobník
- S procesem sdruženy registry a další info potřebné k běhu procesu = stavové informace
  - registry – čítač instrukcí PC, ukazatel zásobníku SP, univerzální registry



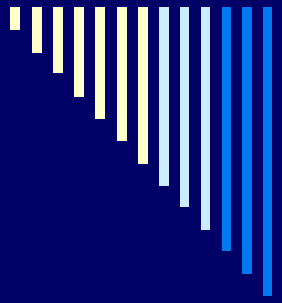
Základní služby OS pro práci s procesy

- **Vytvoření nového procesu**
  - fork v UNIXu, CreateProcess ve Win32
- **Ukončení procesu**
  - exit v UNIXu, ExitProcess ve Win32
- **Čekání na dokončení potomka**
  - wait (waitpid) v UNIXu,
  - WaitForSingleObject ve Win32



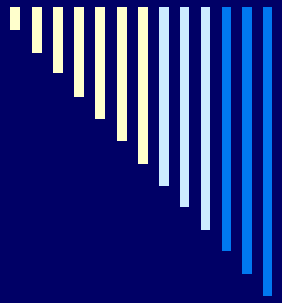
Další služby - procesy

- Alokace a uvolnění paměti procesu
- Komunikace mezi procesy (IPC)
- Identifikace ve víceuživatel. systémech
  - identifikátor uživatele (UID)
  - skupina uživatele (GID)
  - proces běží s UID toho, kdo ho spustil
  - v UNIXu – UID, GID – celá čísla
- Problém uvíznutí procesu



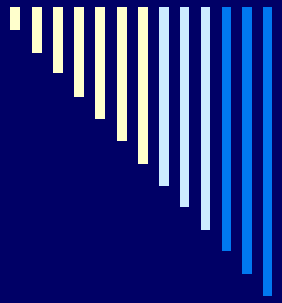
Soubory

- Zakrytí podrobností o discích a I/O zařízení
- Poskytnutí abstrakce – soubor
- Systémová volání
  - vytvoření, zrušení, čtení, zápis
- Otevření a uzavření souboru – open, close
- Sekvenční nebo náhodný přístup k datům
- Logické sdružování souborů do adresářů
- Hierarchie adresářů – stromová struktura



Soubory II.

- Ochrana souborů, adresářů přístupovými právy
  - kontrola při otevření souboru
  - pokud není přístup – chyba
- Připojitelnost souborových systémů
  - Windows – disk určený prefixem *C:, D:*
  - Unix – *kamkoliv* v adresářovém stromu

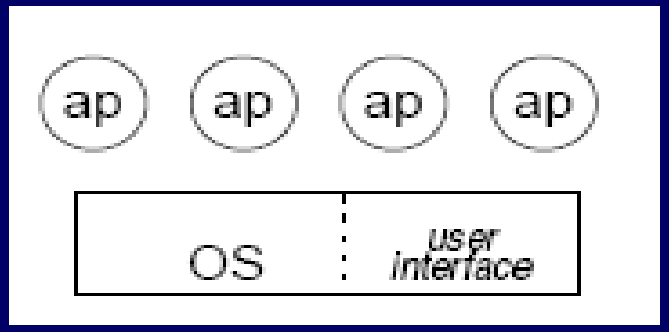


Uživatelské rozhraní

- řádková – CLI (Command Line Interface)
- grafická uživ. rozhraní (GUI)

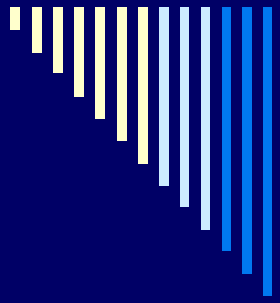
- původně UI součást jádra
- v moderních OS – jedním z programů, možnost náhrady za jiné

UI – obrázky



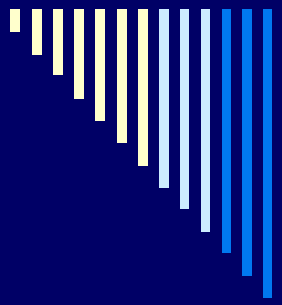
UI jako součást jádra

UI v uživ. režimu



Uživatelské rozhraní - příklady

- GUI Linux
  - systém XWindow (zobrazování grafiky) a grafické prostředí (správci oken,...) – programy v uživatelském režimu
- Windows NT,2000,XP
  - grafická část v jádře
  - logická část (v uživatelském režimu)
  - výkon vs. stabilita

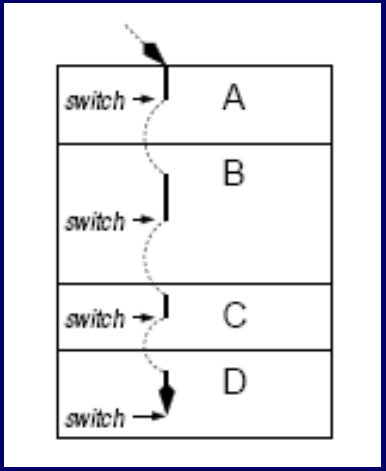


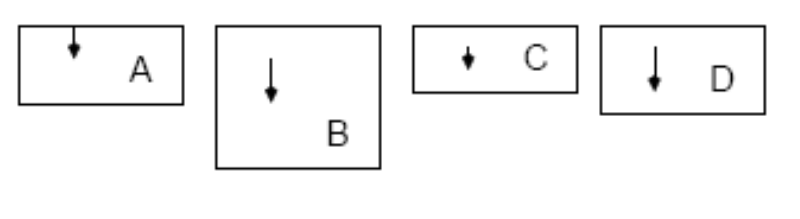
Proces jako abstrakce

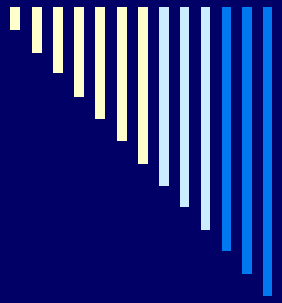
- Běžící SW – organizován jako množina sekvenčních procesů
- Proces – běžící program včetně obsahu čítače instrukcí, registrů, proměnných; běží ve vlastní paměti
- Koncepčně každý proces – vlastní virtuální CPU
- Reálný procesor – přepíná mezi procesy (multiprogramování)
- Představa množiny procesů běžících (pseudo)paralelně

Ukázka

- 4 procesy, každý má vlastní bod běhu (čítač instrukcí) – pseudoparalelní běh x paralelní



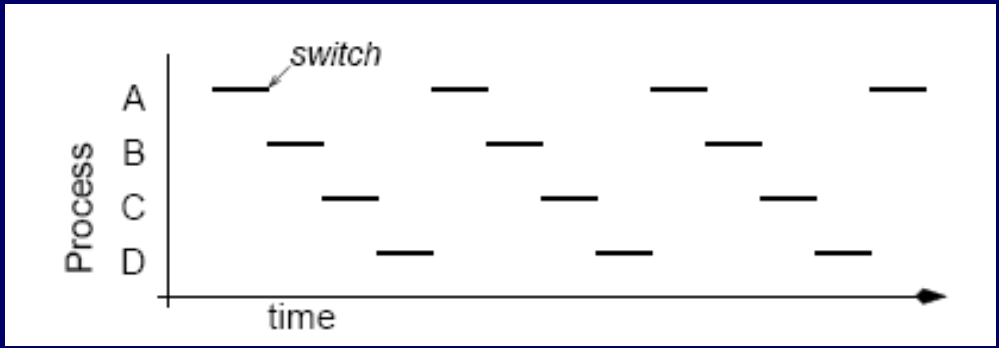


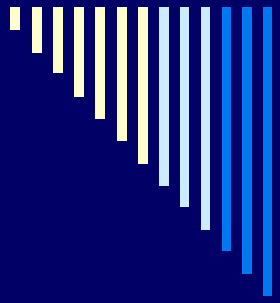


Pseudoparalelní běh

- Pseudoparalelní běh – v jednu chvíli aktivní pouze jeden proces
- Po určité době pozastaven a spuštěn další
- Po určité době všechny procesy vykonají část své činnosti

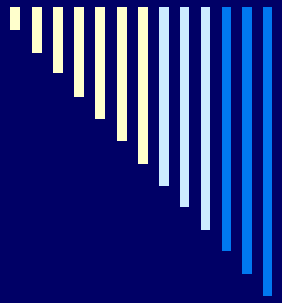
Pseudoparalelní běh





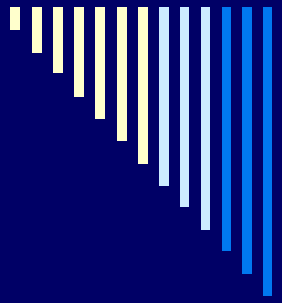
Rychlost procesů

- Rychlost běhu procesu není konstantní.
- Obvykle není ani reprodukovatelná.
- Procesy nesmějí mít vestavěné předpoklady o časování.
- Např. doba trvání I/O různá.
- Procesy neběží stejně rychle.



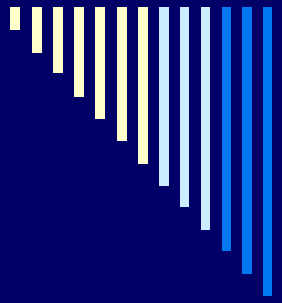
Stavy procesu

- Procesy často potřebují komunikovat s ostatními procesy:
- *ls –l | more*
- Příkaz ls vypíše adresář
- More zobrazí obrazovku a čeká na uživ.
- More je připraven běžet, ale nemá žádný vstup – zablokuje se dokud vstup nedostane



Kdy proces neběží

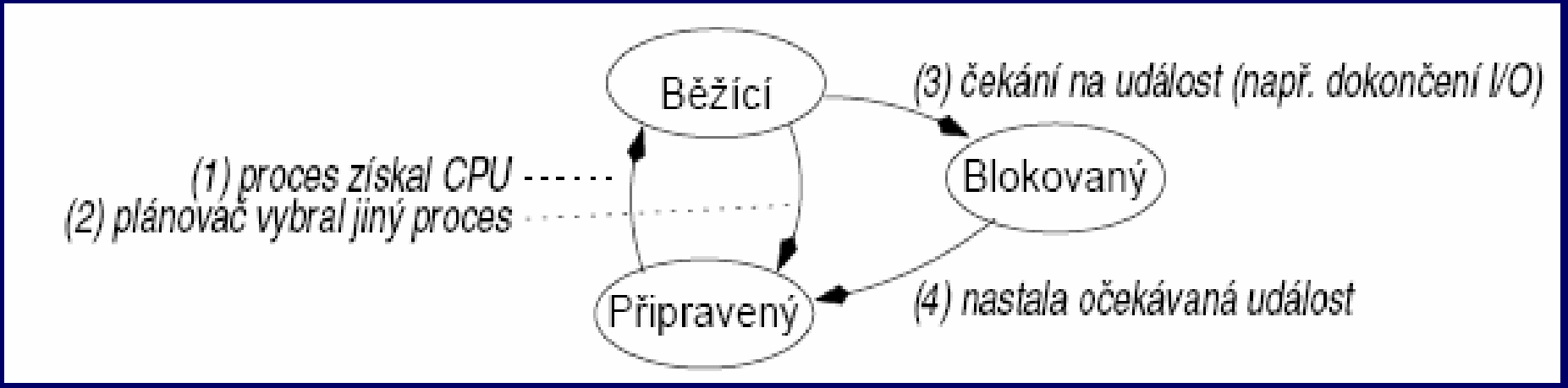
- Blokování procesu – proces nemůže pokračovat, protože čeká na zdroj (vstup, zařízení, paměť), který není dostupný – proces nemůže logicky pokračovat
- Proces může být připraven pokračovat, ale CPU vykonává jiný proces – musí počkat, až bude CPU „volné“

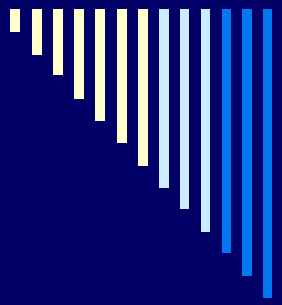


Základní stavy procesu

- **Běžící (running)** – skutečně využívá CPU, vykonává instrukce
- **Připraven (ready, runnable)** – dočasně pozastaven, aby mohl jiný proces pokračovat
- **Blokován (blocked, waiting)** – neschopný běhu, dokud nenastane externí událost

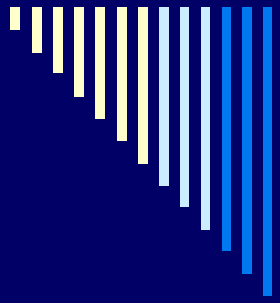
Základní stavy procesu





Přechody stavů procesu

- Plánovač **vybere** tento proces
- Proces je **pozastaven**, plánovač vybere jiný proces
- Proces se **zablokuje**, protože čeká na událost (zdroj – disk, čtení z klávesnice)
- **Nastala** očekávaná **událost**, např. zdroj se stal dostupný

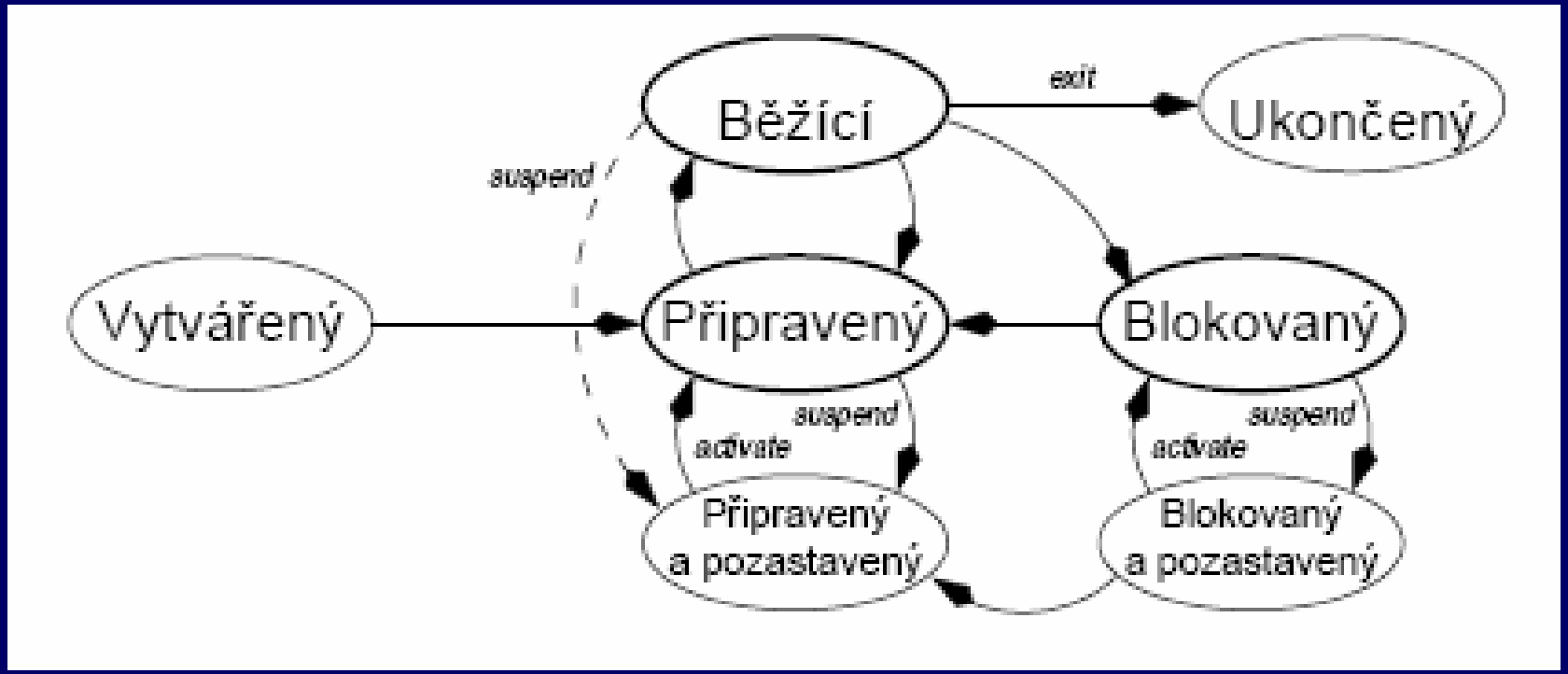


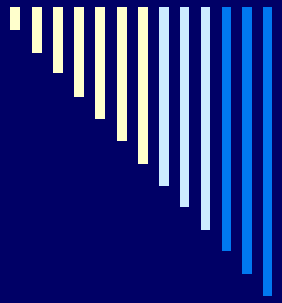
Stavy procesů

- Jádro OS obsahuje plánovač
- Plánovač určuje, který proces bude běžet
- Nad OS řada procesů, střídají se o CPU

- Stav procesu **pozastavený**
- V některých systémech může být proces **pozastaven** nebo **aktivován**
- V diagramu přibudou **dva** nové stavy

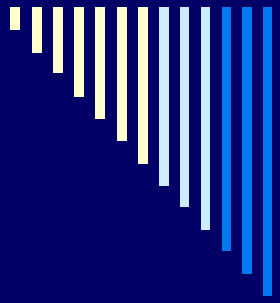
Stavy procesů





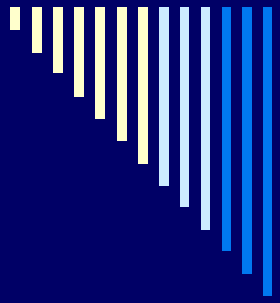
PCB (Process Control Block)

- OS udržuje tabulku nazývanou tabulka procesů
- Každý proces – položku PCB (Process Control Block)
- PCB obsahuje všechny info potřebné pro opětovné spuštění procesu
- Konkrétní obsah PCB – různý
- Pole správy procesů, správy paměti, správy souborů



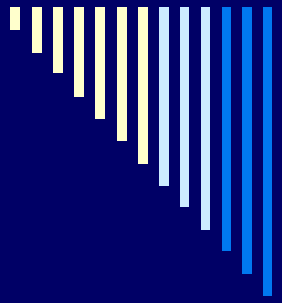
Položky - správa procesů

- Identifikátory (číselné)
  - Identifikátor procesu
  - Identifikátor uživatele
- Stavová informace procesoru
  - Univerzální registry, PC, ukazatel zásobníku SP
  - Stav CPU – PSW (Program Status Word)
- Stav procesu (běžící, připraven, blokován)
- Plánovací parametry procesu (algoritmus, priorita)



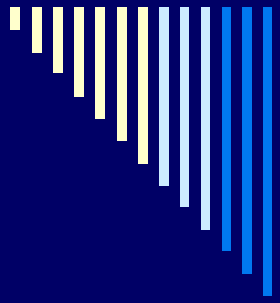
Položky – správa procesů II

- Odkazy na rodiče a potomky
- Účtovací informace
  - Čas spuštění procesu
  - Čas CPU spotřebovaný procesem
- Nastavení meziprocesové komunikace
  - Nastavení signálů, zpráv



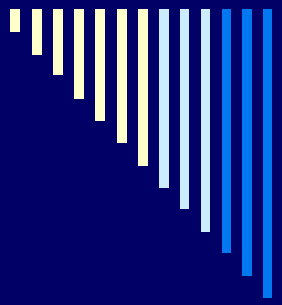
Položky – správa paměti

- Popis paměti
  - Ukazatel, velikost, přístupová práva
- Úsek paměti s kódem programu
- Data – hromada
  - Pascal – new release
  - C – malloc, free
- Zásobník
  - Návratové adresy, parametry funkcí a procedur, lokální proměnné



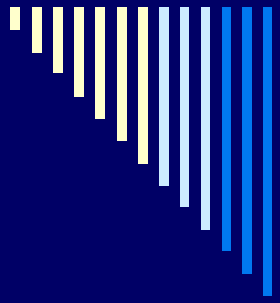
Položky – správa souborů

- Nastavení prostředí
  - Aktuální pracovní adresář
- Otevřené soubory
  - Způsob otevření – čtení / zápis
  - Pozice v otevřeném souboru



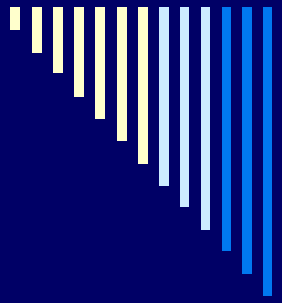
Přepnutí procesu - průběh

- Systém nastaví časovač – pravidelně přerušení
- Na předem definovaném místě – adresa obslužného programu přerušení
- CPU po příchodu přerušení provede:
  - Uloží čítač instrukcí PC do zásobníku
  - Načte do PC adresu obsluž. programu přerušení
  - Přepne do režimu jádra



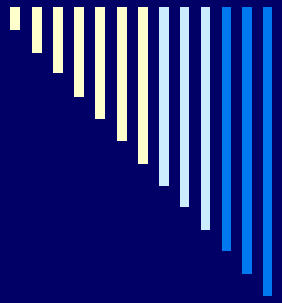
Přepnutí procesu - II

- Vyvolána obsluha přerušení:
  - Uloží obsah registrů do zásobníku
  - Nastaví nový zásobník
- Plánovač nastaví proces jako ready, vybere nový proces pro spuštění
- Přepnutí kontextu
  - Nastaví mapu paměti nového procesu
  - Nastaví zásobník, načte obsah registrů
  - Provede návrat z přerušení – RET (do PC adresa ze zásobníku, přepne do uživatelského režimu)



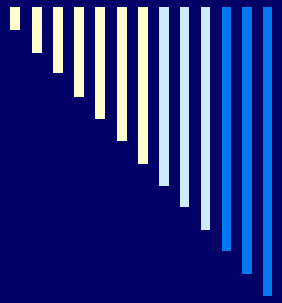
Rychlost CPU vs. paměť

- CPU
  - Rychlost – počet instrukcí za sekundu
  - Obvykle nejrychlejší komponenta v systému
  - Skutečný počet instrukcí závisí na rychlosti, jak lze instrukce a data přenášet z a do hlavní paměti
- Hlavní paměť
  - Rychlost v pamětových cyklech (čtení, zápis)
  - O řád pomalejší než CPU



Rozdíly rychlostí – „pyramida“

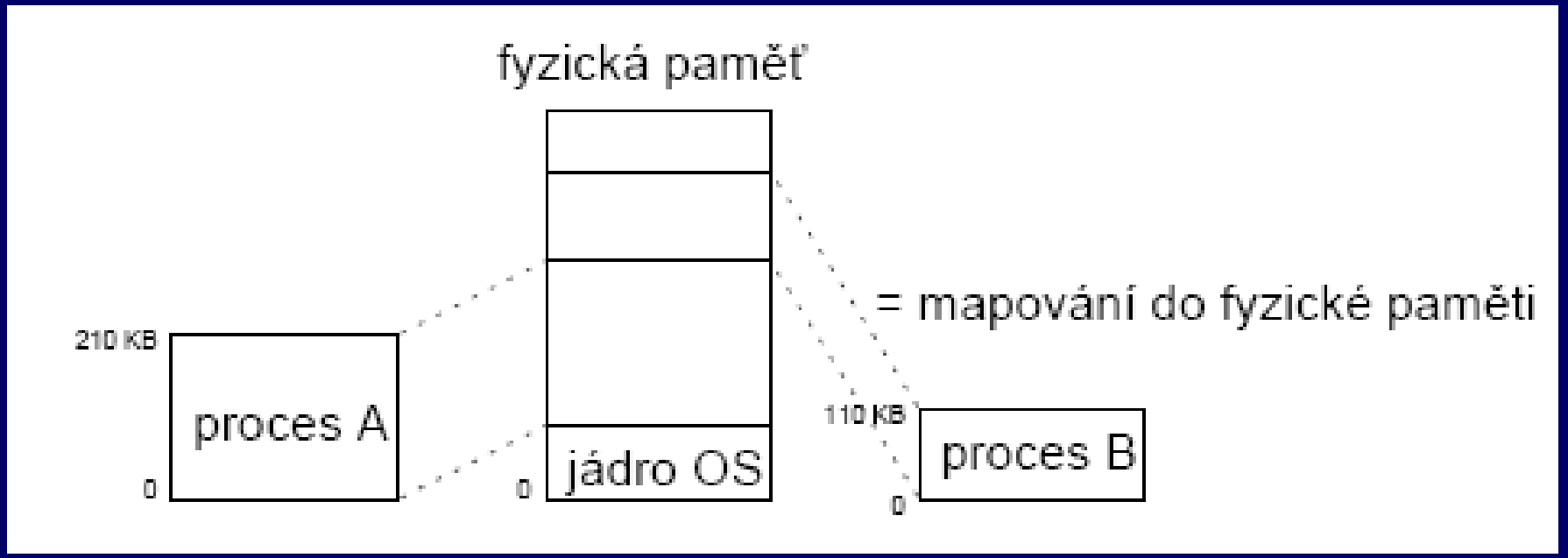
- CPU – rychlé registry – zápisníková pamět, 32x32 nebo 64x64 bitů, žádné zpoždění při přístupu
- Cache – malá paměť s vysokou rychlostí, princip lokality, pokud jsou data v cache – dostaneme velmi rychle, 2 tiky hodin
- Vnější pamět
  - Mechanická, pomalejší, větší kapacita, levnější cena za bit

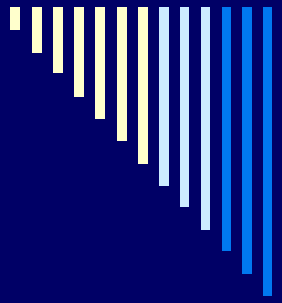


MMU – Memory Management Unit

- Více procesů v paměti
  - Každý proces pamět pro sebe, např. od adresy 0 (relokace)
  - Ochrana – nemůže zasahovat do paměti ostatních procesů ani jádra
- Mezi CPU a pamětí je MMU
  - Program pracuje s virtuálními adresami
  - MMU je převede na fyzické adresy

MMU

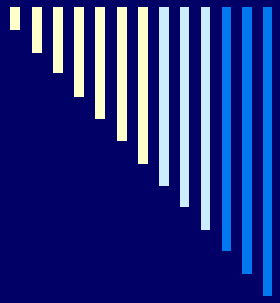

fyzická paměť
= mapování do fyzické paměti
210 KB
proces A
0
jádro OS
0
110 KB
proces B
0



Výkonnostní důsledky

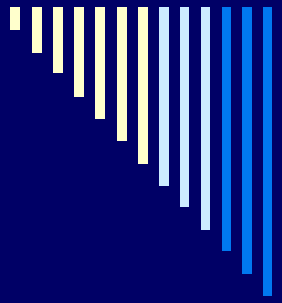
- Pokud program nějakou dobu běží – v cache jeho data a instrukce – dobrá výkonnost
- Při přepnutí na jiný proces – převažuje přístup do hlavní paměti
- Nastavení MMU se musí změnit

- Přepnutí mezi úlohami i přepnutí do jádra (volání služby OS) – relativně drahé (čas)



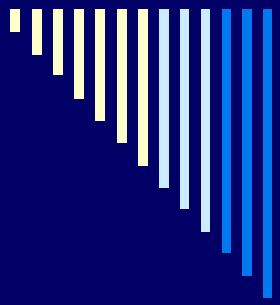
Služby pro práci s procesy

- Jednoduché systémy
  - Všechny potřebné procesy spuštěny při startu systému
  - Běží po celou dobu běhu systému – žádné služby nepotřebujeme
  - Zapouzdřené (embedded) systémy



UNIX a Linux

- Služba **fork()** – vytvoří přesnou kopii rodičovského procesu
- Návratová hodnota – rozliší mezi rodičem a potomkem (potomek dostane 0)
- *pid = fork();*
- *if (pid == 0) potomek else rodic*
- Potomek může činnost ukončit pomocí exit()
- Rodič může na potomka čekat – wait()

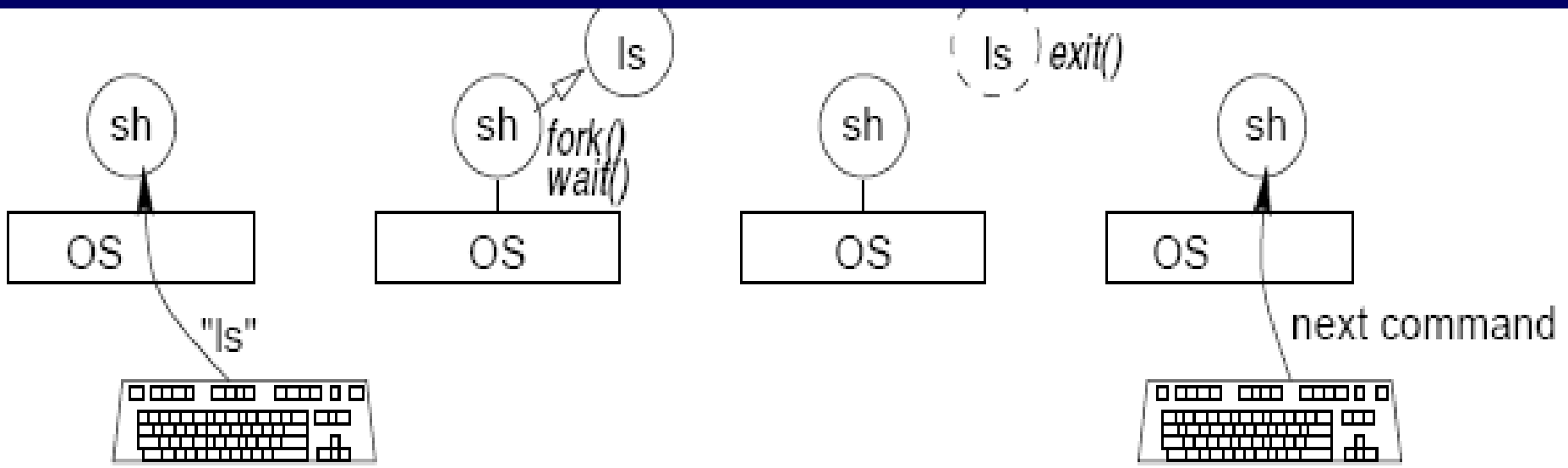


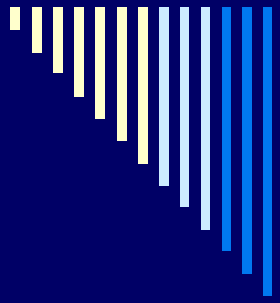
UNIX

- Potomek může místo sebe spustit jiný program – volání execve() – nahradí obsah paměti procesem spouštěným ze zadaného souboru
- *if (fork() == 0)*
- *execve(“/bin/ls”, argv, envp);*
- *else*
- *wait(NULL);*

Příkazový interpret

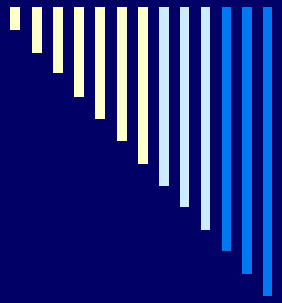
- Spouští příkaz – vytvoří nový proces, čeká na jeho dokončení; ukončení – volání sl. systému





Win32

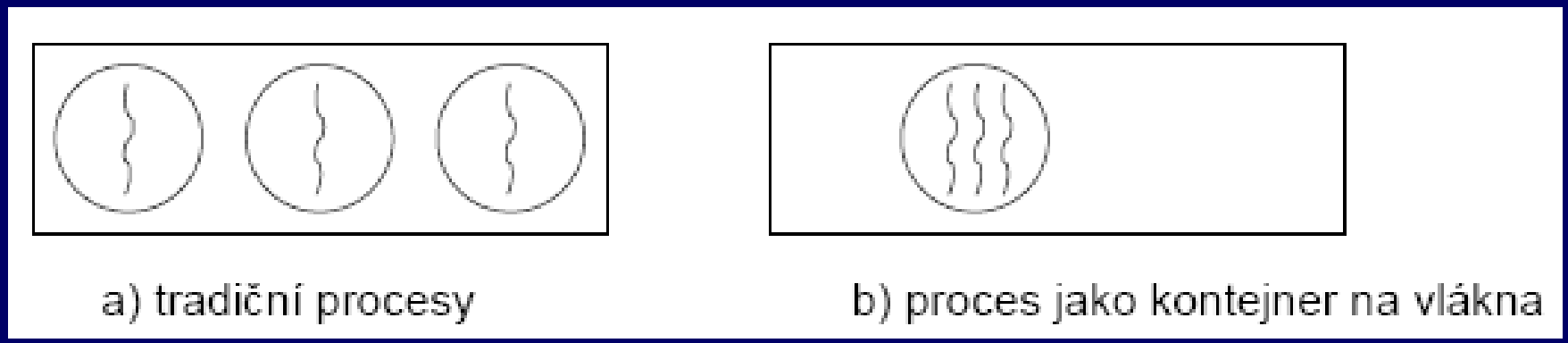
- Vytvoření procesu službou CreateProcess
- Mnoho parametrů – vlastnosti procesu
- Není koncept rodič,potomek – rovnocenné procesy

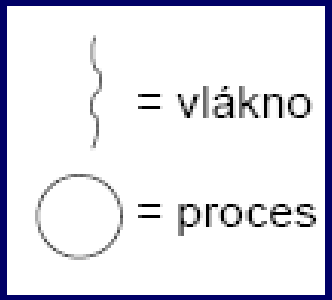


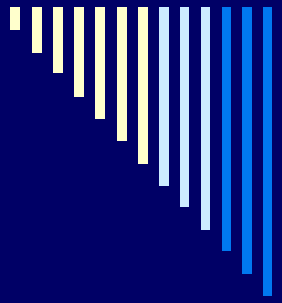
Procesy a vlákna

- Tradiční OS – každý proces svůj vlastní adresový prostor a místo kde běží (bod běhu)
- Často výhodné – více bodů běhu, ale ve stejném adresovém prostoru
- Bod běhu – vlákno (thread, lightwight process)
- Více vláken ve stejném procesu - multithreading

Procesy a vlákna

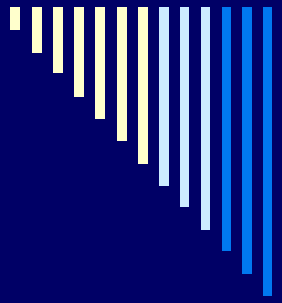






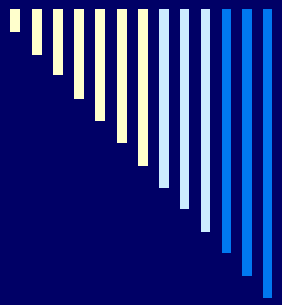
Vlákna

- Vlákna v procesu sdílejí adresní prostor, otevřené soubory (atributy procesu)
- Vlákna mají soukromý čítač instrukcí, obsah registrů, soukromý zásobník
  - Mohou mít soukromé lokální proměnné
- Původně využívána zejména pro VT výpočty na multiprocesorech (každé vlákno vlastní CPU, společná data)



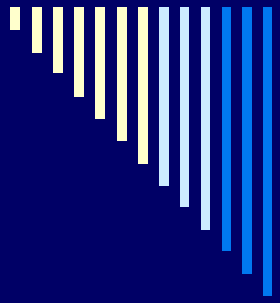
Vlákna – použití dnes

- Rozsáhlejší výpočet a rozsáhlejší i/o
- Interaktivní procesy – jedno vlákno pro komunikaci s uživatelem, další činnost na pozadí
- www prohlížeč – jedno vlákno příjem dat, další zobrazování a interakce s uživatelem
- Textový procesor – vstup dat, přeformátování textu
- Servery www – jedno vlákno pro každého klienta



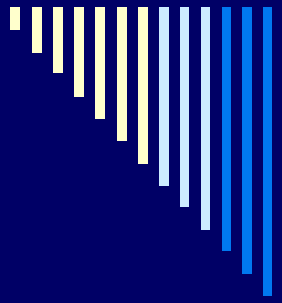
Multithreading

- Podporován většinou OS
  - Linux, Windows
- Často i moderní jazyky
  - Java
- Proces začíná svůj běh s jedním vláknem, ostatní vytváří za běhu programově (konstrukce vytvoř vlákno)
- Režie na vytvoření vlákna a přepnutí kontextu menší než v případě procesů



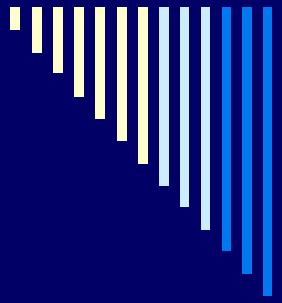
Poznámka

- Jeden proces – více vláken
- Více procesů sdílejících pamět
- Z hlediska např. synchronizace se tyto případy většinou v literatuře nerozlišují



Programové konstrukce pro vytváření vláken

- Statické
  - Proces obsahuje deklaraci pevné množiny podprocesů (např. tabulka)
  - Všechny spuštěny při spuštění procesu
- Dynamické
  - Procesy mohou vytvářet potomky dynamicky
- Pro popis – precendenční grafy

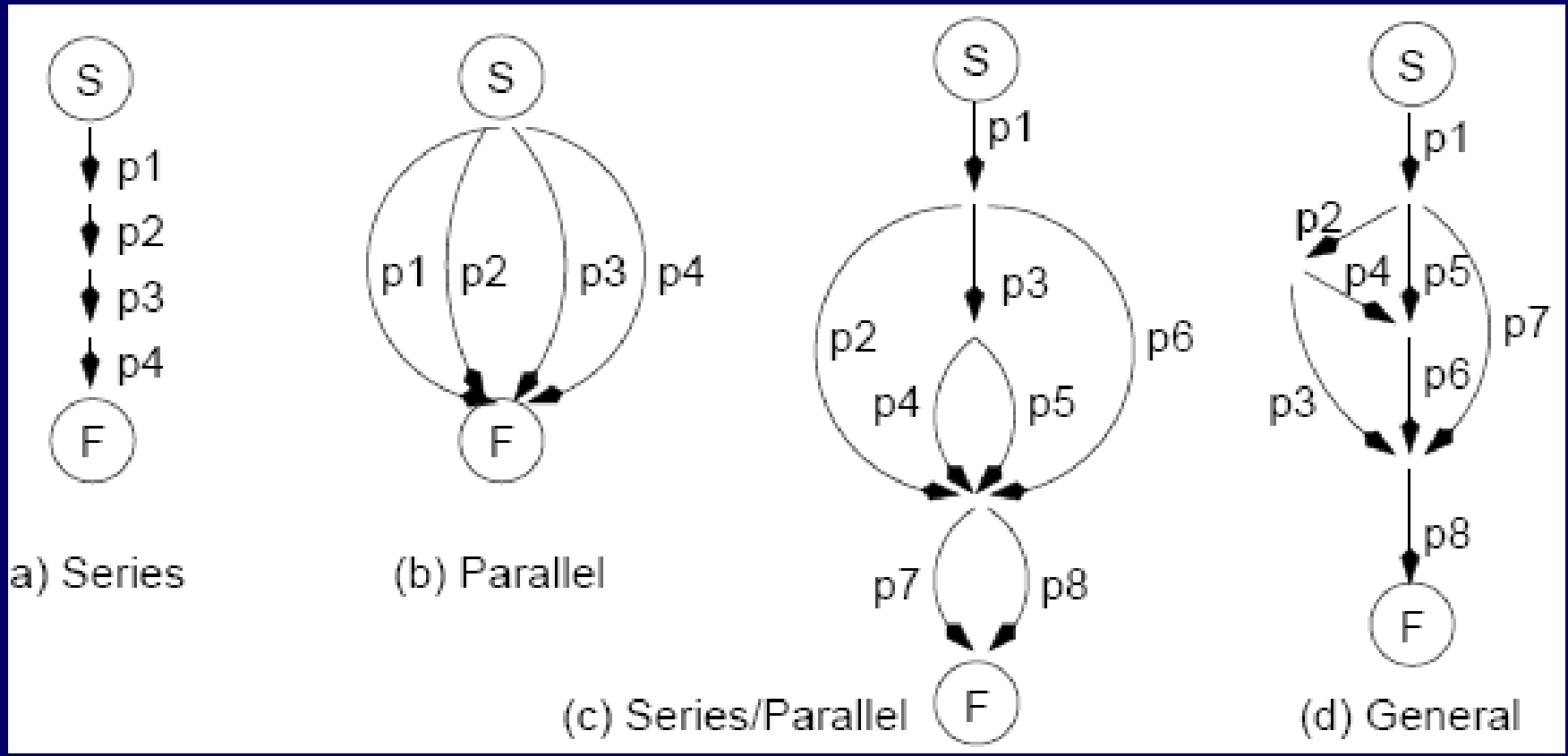


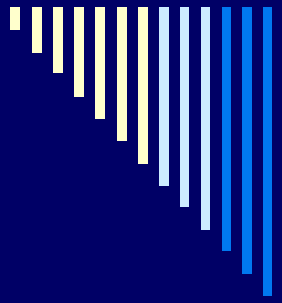
Precedenční grafy

- Popis pro vyjádření různých relací mezi procesy
- Process flow graph

- Acyklický orientovaný graf
- Běh procesu pi – orientovaná hrana grafu
- Vztahy mezi procesy – seriové nebo paralelní spojení – spojením hran

Precedenční grafy





Další materiály

- Dále viz *p2proc.pdf*