



Počítačové sítě a operační systémy

PB169

Počítačové sítě a operační systémy

Jaromír Plhák

xplhak@fi.muni.cz



Organizace výuky

- Přednášky
 - Nepovinné
 - Prezentace jsou dostupné v ISu
 - Ve studijních materiálech
 - 12 přednášek
- Cvičení
 - Povinná*
 - Maximálně dvě neomluvené neúčasti
 - Praktické základy práce s Windows, Linuxem a tvorbou webových stránek

*V případě prokázání dostatečných znalostí nebude nutné cvičení absolvovat

Hodnocení

- Docházka a aktivita na cvičeních
 - Účast na zkušebním termínu je podmíněna dostatečnou docházkou na cvičeních
 - Možnost získat bonusové body za aktivitu na cvičeních (Kahoot)
- Ústní zkouška
 - Dvě otázky
 - OS a sítě, případně bezpečnost
 - Bude přihlédnuto k počtu bonusových bodů



Zkouškové otázky - Operační systémy

- 1. Úvod do OS + Procesy a vlákna
- 2. Plánování CPU
- 3. Správa paměti
- 4. Synchronizace procesů
- 5. Uvážnutí
- 6. I/O systém + Vnější paměti



Zkouškové otázky - Počítačové sítě

- 1. Bezpečnost v informačních technologiích
- 2. Architektura počítačových sítí + Model OSI (obecně)
- 3. Přenos dat v počítačových sítích
- 4. Řízení přístupu k médiu
- 5. Protokoly pro přenos dat v síti + Směrování
- 6. Protokoly aplikační vrstvy + Zabezpečení počítačových sítí



Studijní materiály (1)

- **Prezentace k přednáškám**
 - [Interaktivní osnova k přednáškám](#)
 - [Interaktivní osnova ke cvičením](#)
 - [Záznamy přednášek - jaro 2019](#)
- **Prezentace k předmětům**
 - **PB152 Operační systémy**
 - [Studijní materiály PB152](#)
 - **PB153 Operační systémy a jejich rozhraní**
 - [Studijní materiály PB153](#)
 - **PB156 Počítačové sítě**
 - [Studijní materiály PB156](#)

Studijní materiály (2)

- Prezentace k předmětům
 - PA151 Soudobé počítačové sítě
 - [Studijní materiály PA151](#)
 - PA159 Počítačové sítě a jejich aplikace I
 - [Studijní materiály PA159](#)
 - PA160 Počítačové sítě a jejich aplikace II
 - [Studijní materiály PA160](#)
 - PV169 Základy přenosu dat
 - [Studijní materiály PV169](#)

Studijní materiály (3)

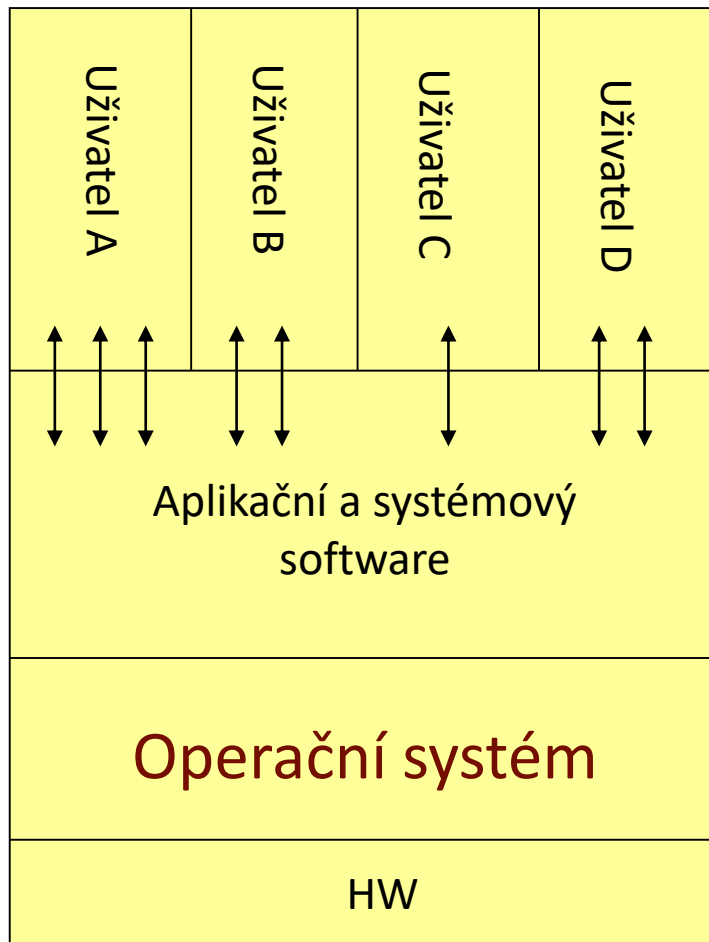
- Silberschatz, Galvin, Gagne: Operating System concepts, 7th edition, Wiley, 2004, ISBN 0-471-69466-5
 - Prezentace k tomuto předmětu jsou založeny na prezentacích k této knize a jsou modifikovány.
 - ©Silberschatz, Galvin and Gagne, 2005
- Stallings: Local and Metropolitan Area Networks, Softcover, Prentice Hall, ISBN 0-13-018653-8



Osnova

- Úvodní hodina
- Procesy a vlákna
- Plánování CPU a Správa paměti
- Synchronizace procesů a Uvážnutí
- I/O systém a Vnější paměti
- Bezpečnost v informačních technologiích
- Architektura počítačových sítí
- Přenos dat v počítačových sítích
- Řízení přístupu k médiu
- Protokoly pro přenos dat v síti a Směrování
- Protokoly aplikační vrstvy a Zabezpečení počítačových sítí
- Anonymní komunikace

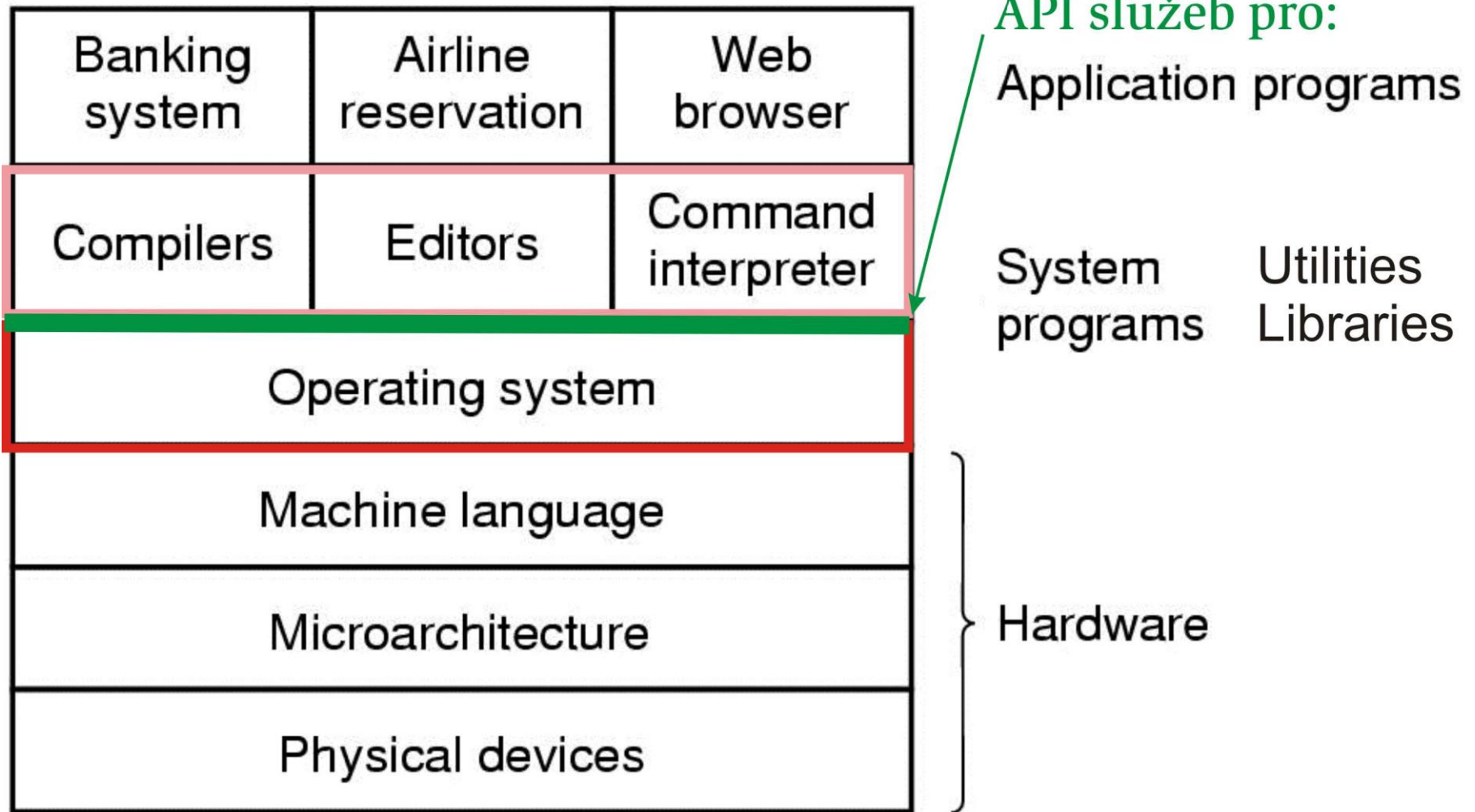
Počítačový systém



- Hardware
 - CPU
 - Paměti
 - I/O
- Operační systém (OS)
- Aplikační a systémový SW
- Uživatelé

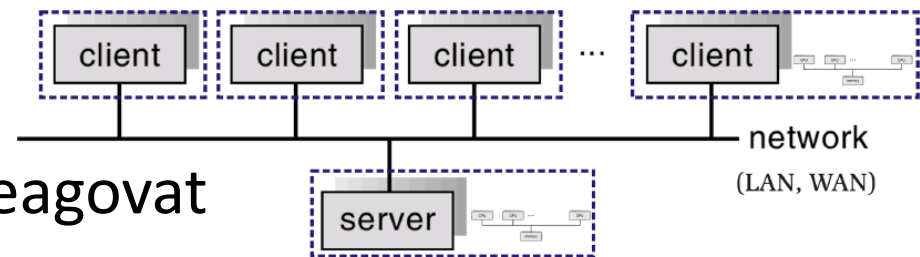
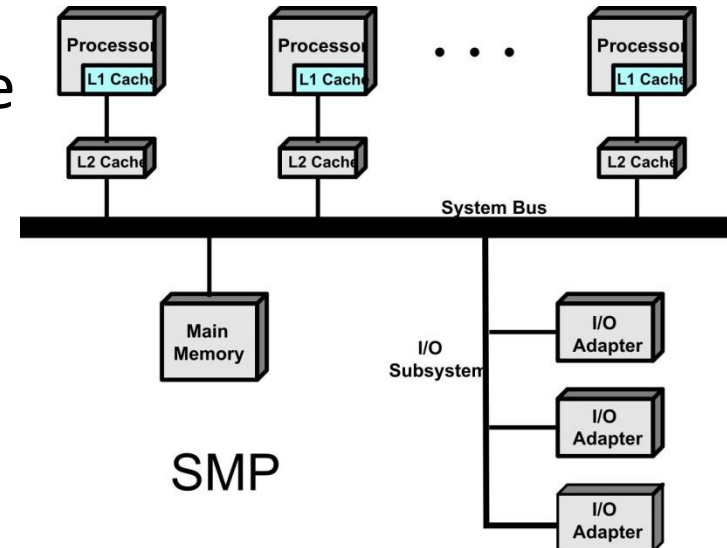


Komponenty počítačového systému



Typy počítačových systémů

- Stolní systémy
 - Vyhrazené pro jednoho uživatele
- Paralelní systémy
 - Sdílí operační paměť a hodinový signál, SMP, AMP
- Distribuované systémy
 - Každý uzel má vlastní paměť
- Real-time systémy
 - Pevně stanovené časové limity kdy musí systém reagovat
- Kapesní systémy



Distribuovaný systém
(typu klient-server)

Proč studovat OS?

- Asi nebudete psát ani navrhovat zcela nový OS ale ...
 - OS je nutné administrovat a efektivně využívat
 - Je nutné rozumět strukturám OS
 - Možná budete muset OS modifikovat nebo rozšiřovat (např. vytvořit nový ovladač)
 - Při programování budete využívat služeb OS
 - Ladění aplikací
 - Techniky používané v OS lze uplatnit i v jiných aplikacích
 - Složité struktury dat, souběžnost, řešení konfliktů, ...
 - A nebo alespoň budete OS používat a je dobré vědět, co od nich můžete čekat

Uživatelský pohled na OS (1)

- Dnes používáme typicky desktopy (či přenosná zařízení) vyhrazené pro jednoho uživatele
 - OS navržen pro jednoduché používání, výkon systému je brán na zřetel, ovšem na využití zdrojů není kladen důraz
 - V současné době je většinou využit multiprocessing
 - Vesměs minimum ochran
 - Hlavní roli hraje uživatelova (administrátorova) zodpovědnost

Uživatelský pohled na OS (2)

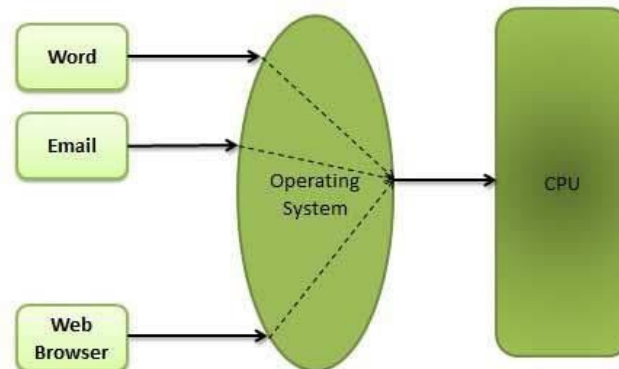
- Dříve často terminály
- OS plní požadavky programů řady uživatelů
 - Důraz na využití zdrojů počítače
 - Férové užívání zdrojů jednotlivými uživateli



DEC VT100 terminal

Systemový pohled na OS

- OS především jako
 - Správce prostředků počítače
 - CPU, operační paměť, disková paměť, I/O zařízení
 - Koordinátor, řídicí složka
 - Řídí spouštění programů, zabraňuje chybám a vzájemnému ovlivňování





Typické oblasti služeb poskytovaných OS (1)

- Vývoj programů, editory, ladící systémy, ...
 - Typicky poskytované pomocnými (systémovými) programy
- Provádění programů
 - Vše co je nutné zajistit pro činnosti řízené programy
 - Plánování, zavádění, ovládání I/O, ...
- Přístup k I/O zařízením
 - Jednotné API pro různá zařízení
- Přístup k souborům dat na vnějších pamětech



Typické oblasti služeb poskytovaných OS (2)

- Přístup k systémovým zdrojům, bezpečnost, řešení konfliktů
- Chybové řízení, automatizované reakce na nestandardní stavy v hardware, v software a v případech kdy OS nemůže uspokojit požadavek aplikace
- Protokolování, informace o tom co se dělo, základ pro účtování, základ pro odhady budoucího vylepšování, ...

Definice OS

- Neexistuje univerzální a všeobecně platná definice OS
- Stejně tak není jednotný názor na to, co všechno zahrnuje OS
 - Jádro, systémové a aplikační programy, ...
 - Definice 1 – OS je to co dá výrobce do krabice
 - Definice 2 – OS je pouze jádro
 - Část, která je neustále spuštěna
- Raději definujeme OS tím co dělá, než tím co vlastně je
- Analogie s „vládou“



Požadavky na OS (1)

- Cíle (povinnosti) OS
 - Řídit řešení uživatelských (aplikačních) programů
 - Poskytnout nástroje pro řešení problémů uživatelů (aplikací)
 - Učinit počítač snadněji použitelný
 - Vytvářet podmínky umožňující efektivně používat hardware počítače

Požadavky na OS (2)

- Cíle (přání) uživatele
 - Služby poskytované OS lze pohodlně používat, snadno zvládnout
 - OS je spolehlivý, bezpečný
 - Požadované služby poskytuje OS pohotově
- Cíle (přání) provozovatele OS
 - OS je snadno navrhnutelný, implementovatelný a udržitelný
 - OS je přizpůsobitelný, spolehlivý a bezchybný

Vývoj OS

- Za 50 let vývoje se OS značně změnily
 - Od jednoduchých textově zaměřených po komplexní systémy s komfortním GUI

```
Current date is Tue 1-01-1980
Enter new date:
Current time is 21:35:24.18
Enter new time:

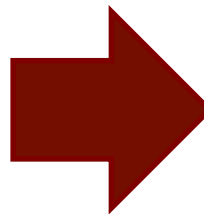
The IBM Personal Computer DOS
Version 2.00 (C)Copyright IBM Corp 1981, 1982, 1983

A>dir

Volume in drive A has no label
Directory of A:\

COMMAND  COM   17664   3-08-83  12:00p
FORMAT   COM    6016   3-08-83  12:00p
CHRDSK   COM    6400   3-08-83  12:00p
SYS      COM    1408   3-08-83  12:00p
DEBUG    COM    11904   3-08-83  12:00p
SLOOP    COM     32    1-01-80   7:44p
        6 File(s)  292864 bytes free

A>_
```



Problémy budování OS

- OS jsou obrovské systémy
 - V současnosti představují až stovky miliónů řádků kódu
 - Pracnost řádově tisíce člověko-roků
- OS jsou složité systémy
 - Požadavky různých uživatelů se často podstatně liší
 - Nelze jednorázově odstranit všechny chyby, verifikace z důvodů složitosti selhává
- Chování OS se obtížně předpovídá, ladění se dělá vesměs odhadem

Multiprogramování (1)

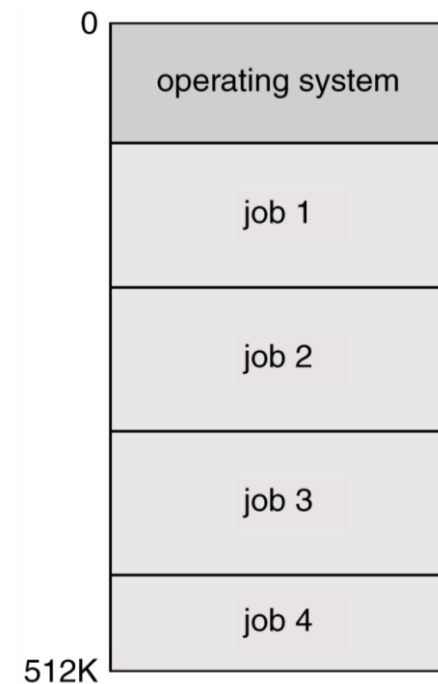
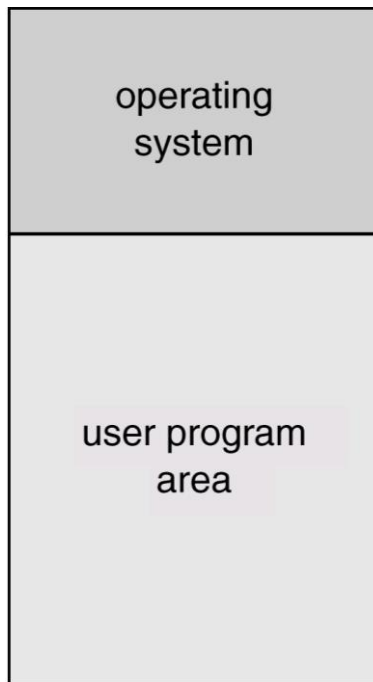
- V paměti je zároveň několik úloh současně
 - Ne však nutně všechny
 - Plánování úloh (job scheduling) – které úlohy umístit do paměti
 - Musíme zajistit ochranu úloh navzájem
- Multiprogramování zvyšuje využití CPU
 - Přidělování CPU jednotlivým úlohám tak, aby CPU byl využit (téměř) vždy

Multiprogramování (2)

- CPU je přidělen úloze, jakmile úloha požádá o I/O operaci, je úloha pozastavena a CPU dostává jiná úloha
 - Pro výběr úlohy, která dostane CPU musí mít CPU plánovací algoritmus
 - Jakmile je I/O operace dokončena, je úloha opět přemístěna do fronty úloh připravených ke spuštění
 - CPU je vytížen, dokud mám úlohy, které nečekají na dokončení I/O operací
 - Dokud úloha nepožádá o I/O operaci, tak má CPU k dispozici
- Jde tedy „pouze“ o efektivnost využití CPU

Obsazení paměti

- V paměti vždy jen jedna úloha
- V paměti několik úloh, běží jen jedna z nich



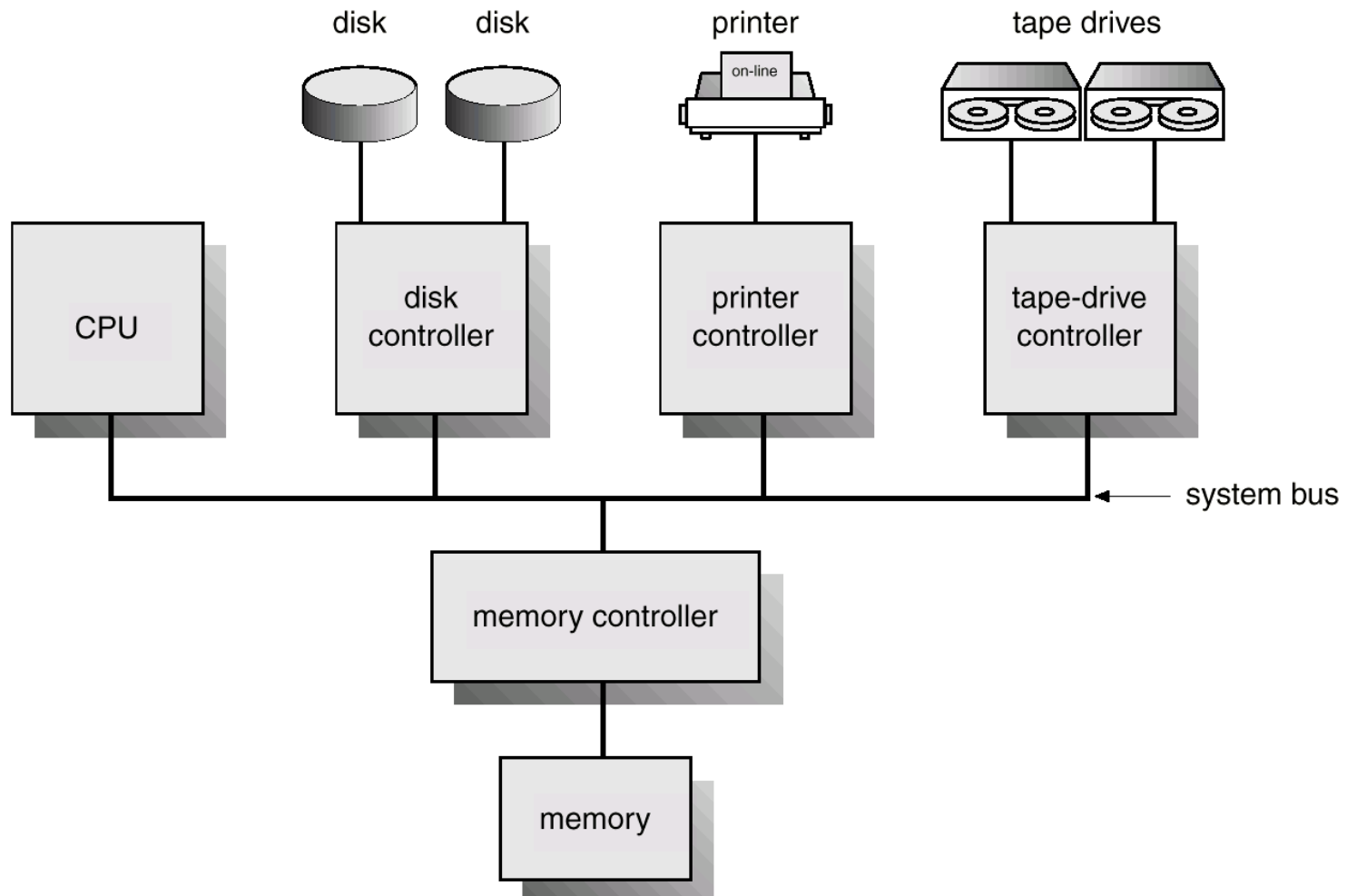
Multitasking

- Time-sharing neboli multitasking
 - Logické rozšíření multiprogramování, kdy úloha (dočasně) ztrácí CPU nejen požadavkem I/O operace, ale také vypršením časového limitu
 - CPU je multiplexován, ve skutečnosti vždy běží jen jedna úloha, mezi těmito úlohami se však CPU přepíná, takže uživatelé získají dojem, že úlohy jsou zpracovávány paralelně

Multitasking (2)

- Multitaskingový systém umožňuje řadě uživatelů počítačový systém *sdílet*
- Uživatelé mají dojem, že počítačový systém je vyhrazen jen pro ně
- Oproti pouhému multiprogramování snižuje dobu odezvy (response time) interaktivních procesů
- Multitaskingové systémy jsou již značně komplexní
 - Správa a ochrana paměti, virtuální paměť
 - Synchronizace a komunikace procesů
 - CPU plánovací algoritmy, souborové systémy

Architektura počítačového systému





Základní vlastnosti počítačového systému

- I/O zařízení typicky mají vlastní vyrovnávací paměť (buffer)
- CPU a I/O zařízení mohou pracovat paralelně
 - Např. řadič disku může ukládat data na disk a CPU může něco počítat
 - Pokud CPU a I/O zařízení pracují paralelně, měla by se nějak synchronizovat
 - Neustálé zjišťování stavu
 - Přerušování

Procesor bez přerušení

- Neustálý cyklus

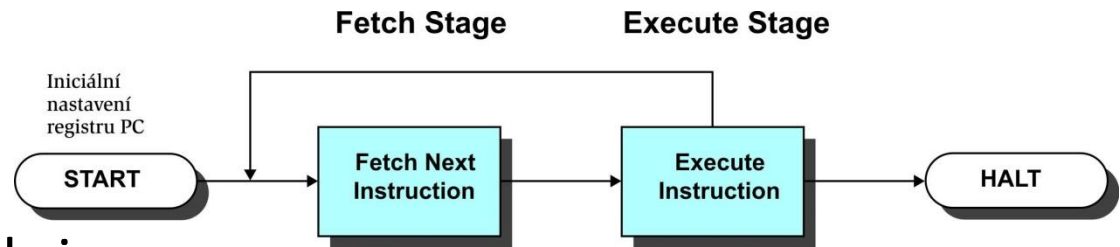
loop

Získej další instrukci

Proveď instrukci

end loop

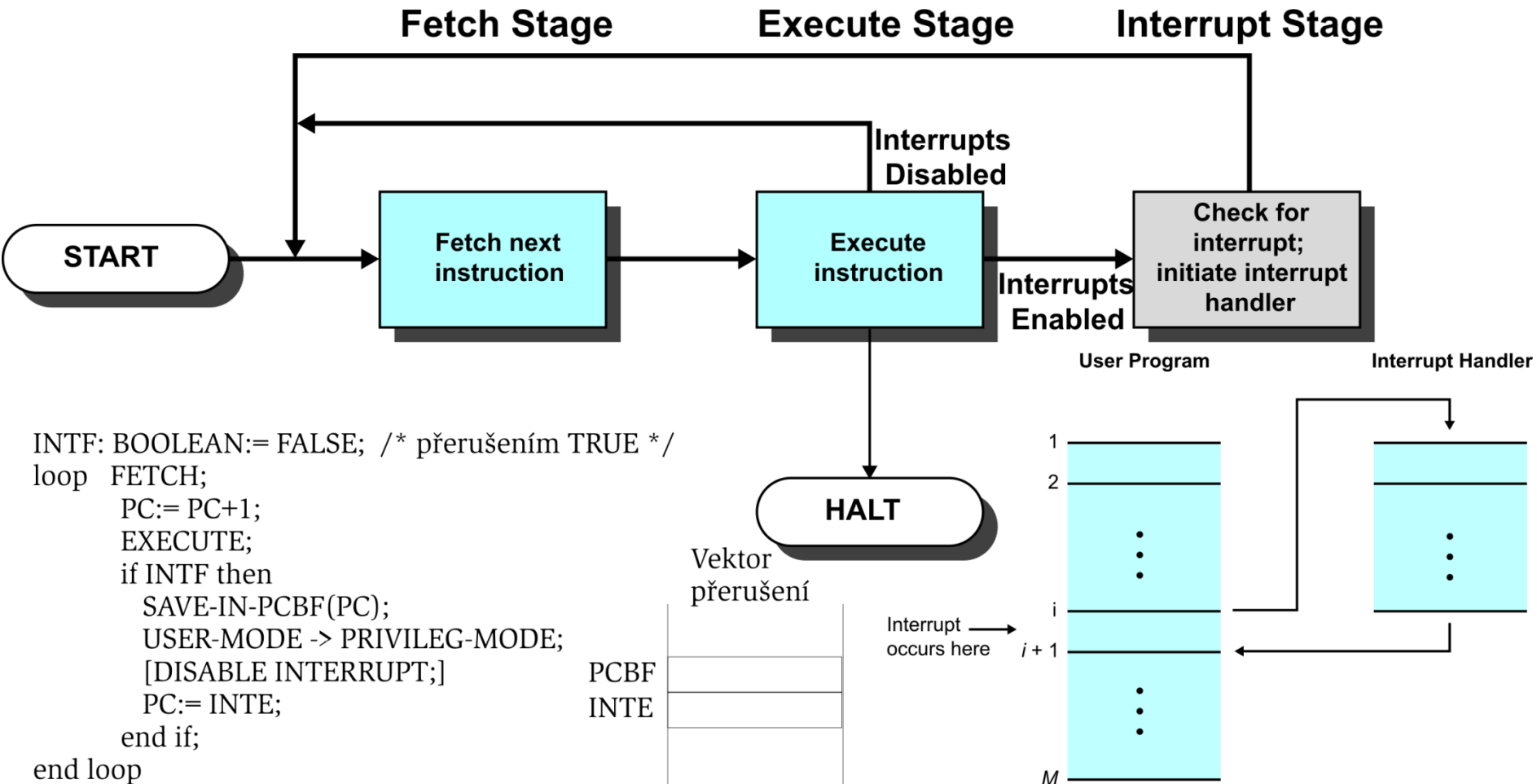
- Pokud chci provést I/O operaci a dozvědět se, že již byla dokončena, musím se periodicky ptát I/O zařízení
- CPU není v těchto chvílích efektivně využit nebo je programování extrémně náročné



Procesor s přerušením (1)

- Základní cyklus je obohacen o kontrolu příznaku přerušení
 - loop
 - Získej další instrukci
 - Proveď instrukci
 - Je-li nastaven příznak přerušení (a přerušení je povoleno), ulož adresu právě prováděného kódu a začni provádět kód rutiny obsluhující přerušení
 - endloop

Processor s přerušením (2)



Přerušení (1)

- Přerušení je signál od I/O zařízení, že se stalo něco, co by OS měl zpracovat
- Přerušení přichází asynchronně
- OS nemusí aktivně čekat na událost
 - Efektivní využití CPU
 - Při výskytu události se automaticky volá patřičná obslužná rutina
 - Umístění obslužných rutin pro jednotlivé typy událostí je typicky dáno tabulkou (v paměti) adres rutin (tzv. vektor přerušení)

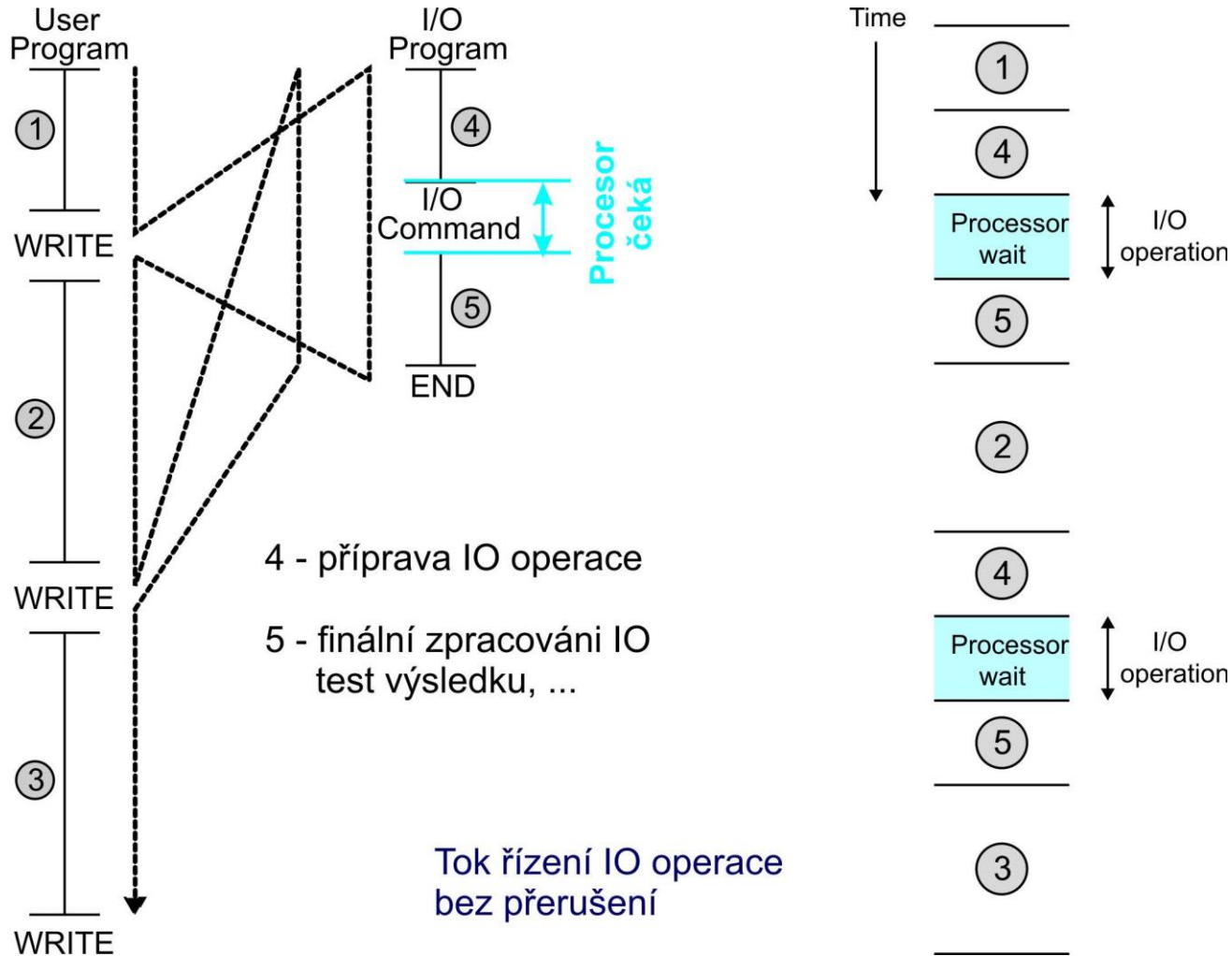
Přerušení (2)

- Aby nedocházelo ke „ztrátě“ přerušení, je při zpracování přerušení další přerušení zakázáno (maskováno)
 - Aby se nepřerušovala rutina obsluhující přerušení
 - Časově náročnější obslužné rutiny však mohou další přerušení explicitně povolit

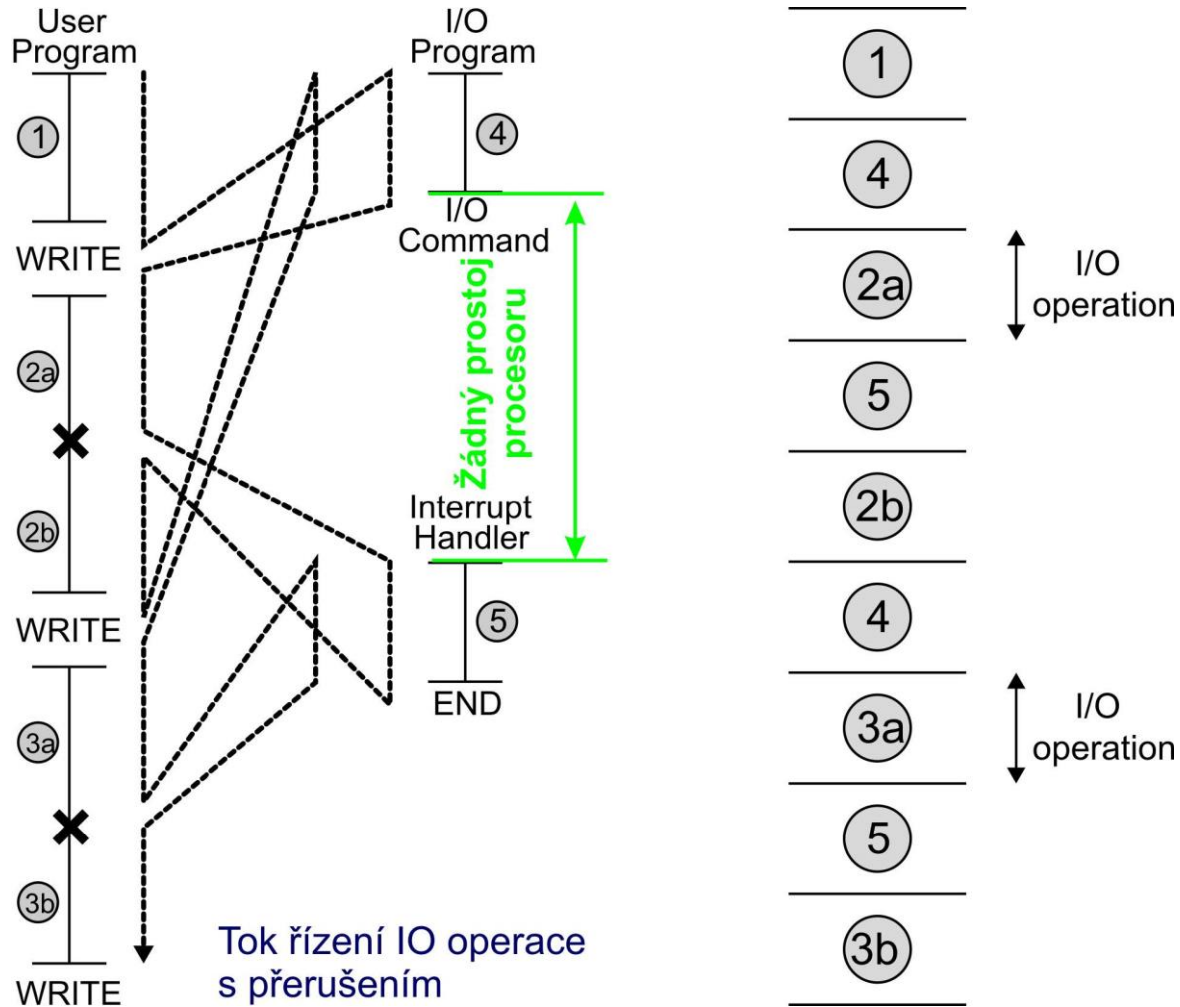
Přerušení (3)

- Operační systém je řízený přerušeními
 - CPU zpracovává uživatelský proces, při příchodu přerušení je spuštěna obslužná rutina OS
 - Návratovou adresu ukládá procesor
 - Hodnoty použitých registrů ukládá ovládací rutina
 - Přerušení nemusí být generováno jen HW, přerušení je možné vyvolat i softwarovými prostředky
 - Tzv. „trap“ – jde vlastně o synchronní přerušení
 - Chyby – dělení nulou
 - Speciální instrukce (např. INT – typicky slouží k systémovému volání)

Tok řízení I/O operací bez prerušení



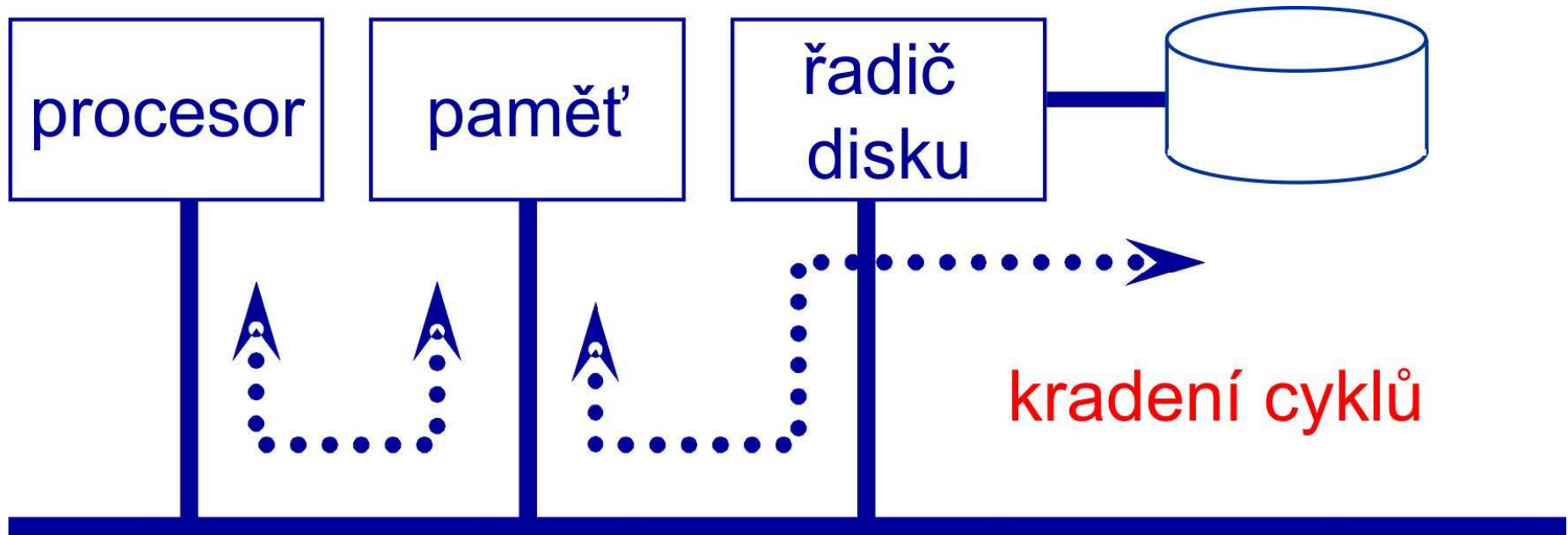
Tok řízení IO operací s prerušením



DMA, Direct Memory Access (1)

- Způsob jak rychle přenášet data mezi I/O zařízením a pamětí
- CPU nemusí přenášet data po jednom bajtu, požádá řadič o přenos celého bloku dat
- CPU požádá o přenos dat, po skončení přenosu dat se o tom dozví pomocí přerušení (tj. 1 přerušení/blok dat ne 1 přerušení/bajt-slovo)
- V době přenosu dat soupeří I/O řadič a CPU o přístup do paměti (oba pracují se „stejnou“ pamětí)

DMA, Direct Memory Access (2)



Ochranné funkce HW

- Je-li v paměti několik procesů, nechceme, aby se procesy mohly navzájem negativně ovlivňovat
 - Přepisování paměti
 - Nespravedlivé získání času CPU
 - Souběžný nekoordinovaný přístup k I/O prostředkům
- Proto OS musí tomuto ovlivňování zabránit
 - Často to však nemůže zajistit sám a potřebuje podporu HW – např. při ochraně přístupu do paměti, přístupu k I/O portům

Režimy procesoru

- Běžný způsob ochrany je dvojitý režim činnosti procesoru
 - Uživatelský režim
 - Privilegovaný režim
- Některé instrukce je možné provést jen v privilegovaném režimu
 - Např. instrukce pro I/O, nastavování některých registrů (např. některé segmentové registry)
- Z privilegovaného režimu do uživatelského režimu se CPU dostane speciální instrukcí, z uživatelského režimu do privilegovaného režimu se CPU dostává při zpracování přerušení

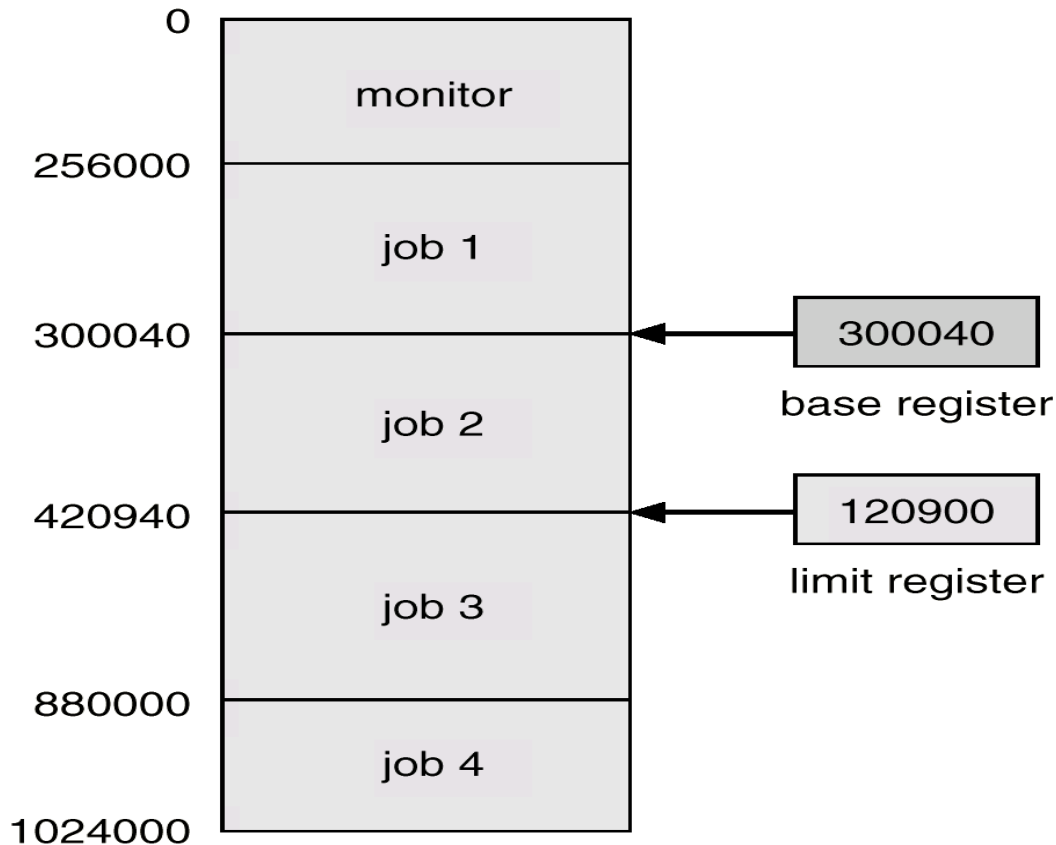
Ochrana paměti

- Minimálně musíme chránit vektor přerušení a rutiny obsluhy přerušení
 - Jinak by bylo možné získat přístup k privilegovanému režimu procesoru
- Každému procesu vyhradíme jeho paměť, jinou paměť nemůže proces používat
 - Ochranu zajišťuje CPU na základě registrů/tabulek nebo principů nastavených OS
 - Např. báze a limit – proces má přístup jen k adresám v rozsahu od (báze + 0) až po (báze + limit)
 - Přístup k nepovoleným adresám způsobí přerušení – to zpracovává OS a např. ukončí činnost procesu

Ochrana CPU

- Jak zaručit, že vládu nad procesorem (tj. jaký kód bude CPU vykonávat) bude mít OS?
- Časovač
 - Časovač generuje přerušení
 - Přerušení obsluhuje OS
 - Ten rozhodne co dál
 - Např. odebere jednomu procesu, vybere další připravený proces a ten spustí (změní kontext)
 - Časovač může generovat přerušení pravidelně nebo je příchod přerušení programovatelný (privilegovanou instrukcí)

Báze a limit



- Jednoduché na implementaci
 - Dva registry, jejichž nastavování je privilegovanou operací
- CPU kontroluje, zda adresy, které proces používá spadají do rozsahu daného registry
- Jak řešit požadavek procesu o přidělení dalšího bloku paměti?



Komponenty OS

- Správa procesů
- Správa operační paměti
- Správa souborů
- Správa I/O zařízení
- Správa sekundární paměti
- Správa síťových služeb
- Ochranný systém
- Interpret příkazů (shell)

Systemová volání

- Systemová volání tvoří rozhraní mezi uživatelským procesem a OS
 - Typicky jsou popsána jako instrukce assembleru a jsou uvedena v programátorském manuálu k OS
 - Vyšší programovací jazyky obsahují některé funkce, které odpovídají systémovým voláním (např. open, write) a dále knihovní funkce, které poskytují vyšší funkčnost a v rámci této spouští (třeba hned několik) systémových volání (např. fopen, fwrite).

Systemová volání (2)

- Různé OS a různé HW platformy mívají různé způsoby jak volat služby OS a různou strukturu těchto služeb
- Nicméně existují určité standardy, které usnadňují přenositelnost programového kódu
 - V oblasti UNIXu – POSIX
 - V oblasti Windows – Win32 (Windows API)
- Teoreticky kód, který bude psán podle standardu, bude přeložitelný na kompatibilních platformách
 - V praxi však existuje celá řada verzí standardu a mnoho výjimek co je a není implementováno

Způsob předání parametrů

- Registry

- `mov eax, 1` // Naplní registr EAX hodnotou 1
- `mov ebx, eax` // Zkopíruje obsah registru EAX do registru EBX
- `int 10h` // Generuj přerušení

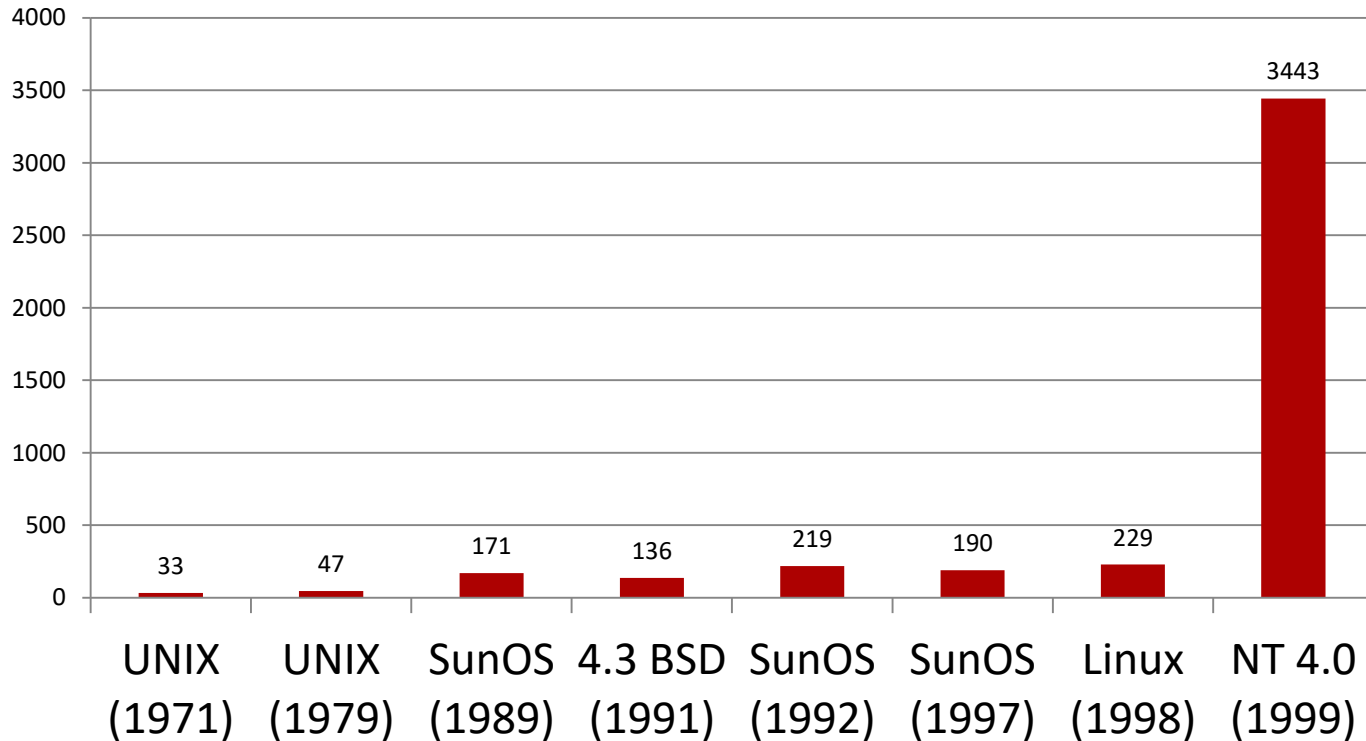
- Zásobník

- `mov ax, 0001h` // Naplní registr ax hodnotou z paměti 0001h
- `push ax` // Vloží ax na vrchol zásobníku
- `int 10h` // Generuj přerušení

Příklad – Linux

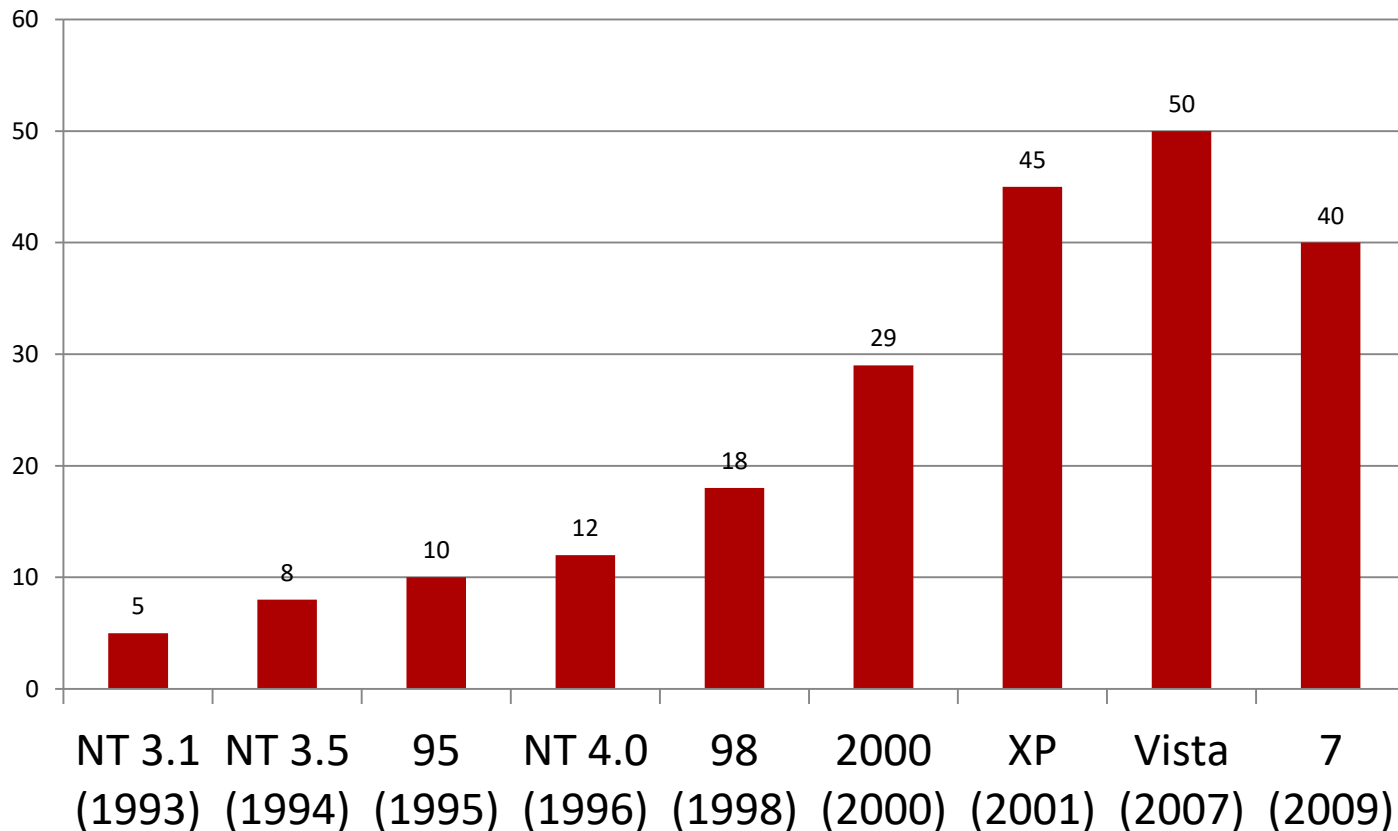
- Systémová volání
 - Standardně přes `int 0x80`
 - Nově i přes instrukci `syscall` (`sysenter`)
- Číslo systémového volání
 - V registru `eax`
 - V jádře 2.2 přibližně 200 volání, v jádře 2.6 přes 300 volání
- Parametry systémového volání
 - V registrech `ebx`, `ecx`, `edx`, `esi`, `edi`, `ebp`
 - Kromě funkce 117 (parametrem je odkaz na strukturu)
- Výsledek
 - Uložen v `eax`

Komplexita OS (1)



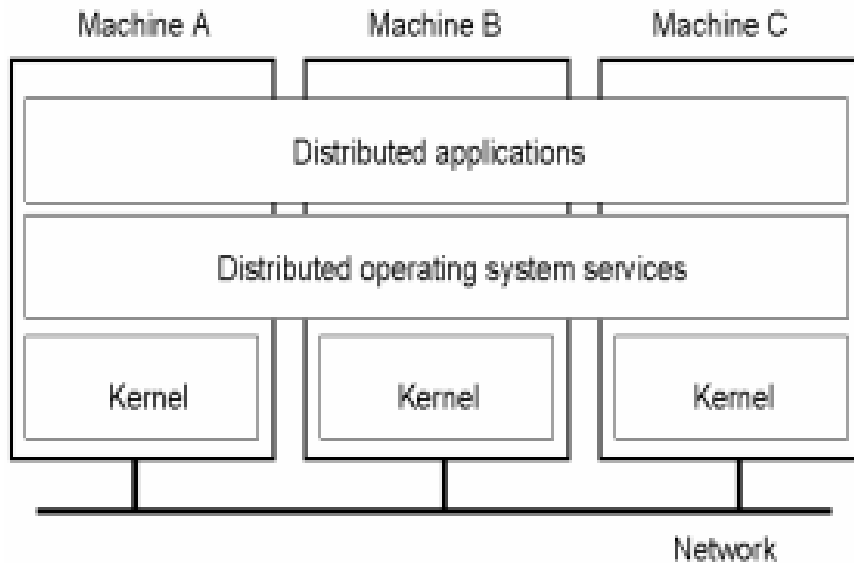
Počet systémových volání OS / API volání

Komplexita OS (2)



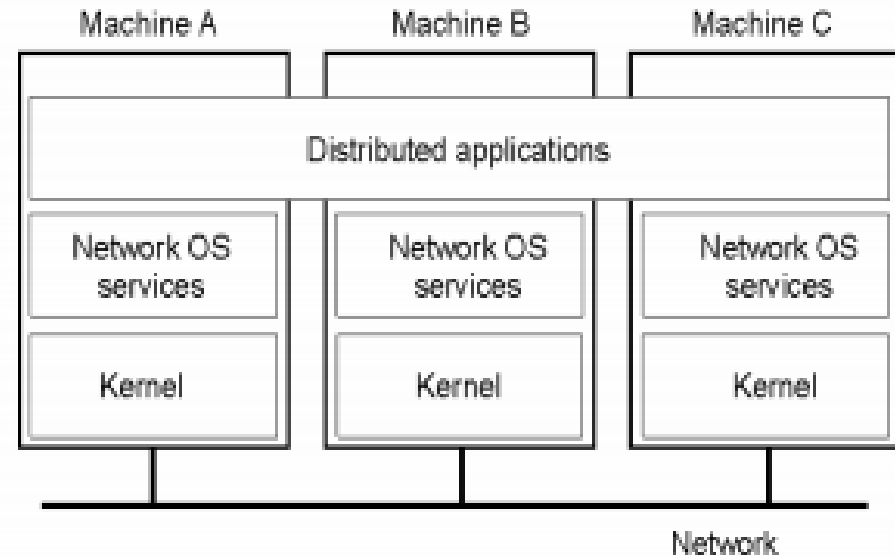
- Rozsah zdrojového kódu Windows (v milionech řádků)

Distribuovaný vs. síťový OS



Distribuovaný OS

- Celý systém se tváří jako tradiční jednoprosesorový systém, přestože jeho jednotlivé komponenty jsou fyzicky rozmístěny na jednotlivých počítačích



Síťový OS

- Každý počítač svůj operační systém
- Uživatelé jsou si vědomi více počítačů
- Mohou kopírovat soubory z jednoho počítače na druhý apod.