

# Operační systémy a sítě

Petr Štěpán, K13133

KN-E-129

[stepan@fel.cvut.cz](mailto:stepan@fel.cvut.cz)

Téma 1. Úvod do přednášky

# Obsah

## Operační systémy – Lekce 1-9

- Silberschatz A., Galvin P. B., Gagne G.: Operating System Concepts  
<http://codex.cs.yale.edu/avi/os-book/OS7/os7c/index.html>
- Tanenbaum A. S.: Modern Operating Systems  
<http://www.cs.vu.nl/~ast/books/mos2/>
- YouTube lectures (anglicky):
  - CS 162 – UC Berkeley
  - OS-SP06 – Surendar Chandra – UC Berkeley
  - MIT 6.004

## Sítě – Lekce 10-13

- Comer D. E.: Internetworking With TCP/IP Volume 1: Principles Protocols, and Architecture,  
[http://www.cs.purdue.edu/homes/dec/vol1/vol1\\_presentation.pdf](http://www.cs.purdue.edu/homes/dec/vol1/vol1_presentation.pdf)
- Jiří Peterka, přednášky na MFF UK, Počítačové sítě, Rodina protokolů TCP/IP  
[http://www.earchiv.cz/i\\_prednmffuk.php3](http://www.earchiv.cz/i_prednmffuk.php3)

# Organizace přednášky

- Souhrnná literatura v češtině není
- Tyto prezentace (stránka předmětu, postupně přidávány  
<https://cw.felk.cvut.cz/wiki/courses/a4b33oss>)
- Cvičení částečně seminární a samostatná práce
  - Odkaz na cvičení ze stránky předmětu
  - Vedoucí cvičení: Ing Jan Chudoba
- Zkouška:
  - Výsledky cvičení (až 10 b.)
  - Test u zkoušky (až 5 b.)
  - Zadané 2 příklady/úlohy (až 15 b.)
  - Hodnocení:
    - $\geq 27$  & Minimálně 9 bodů ze cvičení → A (výborně)
    - 24 – 26,9 & Minimálně 7 bodů ze cvičení → B (velmi dobře)
    - 21 – 23,9 → C (dobře)
    - 18 – 20,9 → D (uspokojivě)
    - 15 – 17,9 → E (dostatečně)

# Cíle předmětu

Podle Hospodářských novin se Informatika vyučuje nejlépe na FEL, ČVUT (22.1.2015)

- OS patří k základům informatiky
- Poznat úkoly OS a principy práce OS
- Využívat OS efektivně a bezpečně
- Seznámit se principy počítačových sítí (reakrediacce OI rozdělila již OS a sítě)

Co **NENÍ** cílem tohoto předmětu

- Naučit Vás jak napsat aplikaci pod (X)Windows
- Naučit triky pro konkrétní OS
- Vytvořit OS – na to je málo času

# Proč studovat OS

- Pravděpodobně nikdo z nás nebude psát celý nový OS
- Proč tedy OS studovat?
  - Každý ho používá a jen málokdo ví jak pracuje
  - Jde o nejrozsáhlejší a nejsložitější IT systémy
  - Uplatňují se v nich mnohé různorodé oblasti
    - softwarové inženýrství,
    - netradiční struktury dat,
    - sítě, algoritmy, ...
  - Čas od času je potřeba OS upravit
    - pak je potřeba operačním systémům rozumět
    - psaní ovladačů, ...
  - Techniky užívané v OS lze uplatnit i v jiných oblastech
    - neobvyklé struktury dat, krizové rozhodování, problémy souběžnosti, správa zdrojů, ...
    - mnohdy aplikace technik z jiných disciplín (např. operační výzkum)
    - naopak techniky vyvinuté pro OS se uplatňují v jiných oblastech (např. při plánování aktivit v průmyslu)

Malý návod na použití školy...

# Cíle vzdělávání

- V obecné rovině
  - Naučit kriticky myslet
  - Naučit hledat zákonitosti
- V konkrétní rovině
  - Předat nějaké konkrétní znalosti (co je posix, cache, sběrnice)
  - Předat nějaké konkrétní dovednosti (jak se programuje, jak efektivně vést projekt)

# O dobrém a špatném učení se...

- Povrchní přístup k učení
  - Úkoly dělám, abych splnil jejich zadání a dostal body
  - Výsledkem je zpravidla memorování
- Hlubkový přístup k učení
  - Úkoly dělám, abych splnil jejich účel
  - Výsledkem je zpravidla porozumění
  - Navíc je nutné najít účel úloh



# Proč porozumět a ne memorovat...

- Schopnost spojit nové a dřívější znalosti
  - Pomáhá v chápání nových znalostí
  - Pomáhá odstranit chybné znalosti
- Schopnost použít znalosti
  - Znalosti lze spojit s každodenní zkušeností
- Schopnost uchovat znalosti
  - Dobře spojené a pochopené znalosti se pamatují déle
- Volba je na Vás !

# Co bude na přednáškách...

- Výkladu se nedá uniknout
  - Náplň je většinou předem k dispozici
- Je to jako kino, ne? Návštěva kina je:
  - Pasivní zážitek s občas zajímavým příběhem
  - Nemusíte příliš přemýšlet
  - Desítky miliónů \$ vynaložené na udržení Vaší pozornosti
- Aktivní učení
  - Charles Lin: Active Learning in the Classroom
  - <http://www.cs.umd.edu/class/sum2003/cmsc311/Notes/Learn/active.html>

# Co bude na přednáškách...

- Když chodíte na přednášky
  - očekává se, že se něco naučíte
- V čem je problém
  - Látka je složitá, ale při poslouchání to člověku nepříjde
  - Většina věcí se jeví logická – myšlenkové zkratky
  - Pro zvládnutí je nutné se jí nějakou dobu věnovat i po přednášce
    - ACM/IEEE CS Curriculum: na 1 hodinu přednášky v bakalářském studiu připadají 2-3 hodiny domácí přípravy

# Jak se něco na přednášce naučit?

- Neusnout
  - Bez ohledu na to, jak těžké to může být
  - Kdo spí, ten se nic nenaučí a přichází o souvislosti
- Chodit pravidelně
  - Nová látka staví na předchozích základech
  - Naučíte se lépe rozumět přednášejícímu
  - Když jsem minule nebyl, alespoň si přečíst přednášky
- Aktivně poslouchat
  - Nejlépe se nové věci naučíte při hledání vlastního vysvětlení, jak věci fungují
  - Dává smysl to co slyšíte?
  - Byli byste schopni to vysvětlit někomu, kdo na přednášce nebyl?
- Pokud něco nedává smysl
  - Zapište si co Vám nedává smysl
  - Zkuste vymyslet otázku, jejíž zodpovědění by věci vyjasnilo a položte ji přednášejícímu

# Kdy a jak se ptát?

- Když Vaše představa neodpovídá tomu co slyšíte
  - Nebo když Vám chybí část „skládanky“
  - Na konci přednášky byste měli být schopni položit několik otázek, alespoň upřesňujících
    - „Myslím si, že říkáte ..(vlastními slovy).., je to tak?“
- Než se zeptáte, zkuste si odpovědět
  - Pokud nevíte, nebo si nejste jisti, zeptejte se
- Při hledání otázek začnete pozorněji poslouchat
  - Začnete poslouchat s cílem se něco naučit
  - Naučíte se klást užitečné dotazy

Konec návodu na použití školy.



# Co je operační systém?

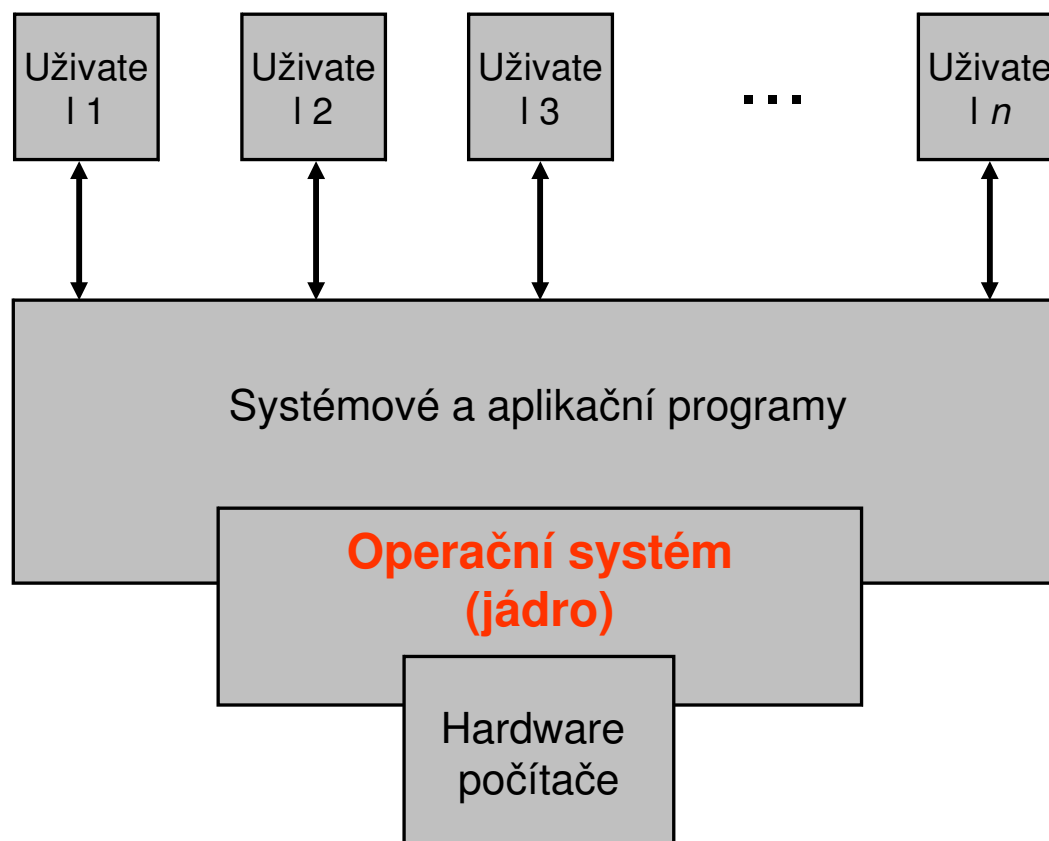
- Neexistuje žádná obecně platná definice
- Několik koncepcí pojmu OS
  - systémové (jen jádro a s ním související nadstavby)
  - „obchodní“ (to, co si koupíme pod označením OS)
  - organizační (včetně pravidel pro hladký chod systému)
- OS jako rozšíření počítače
  - Zakrývá komplikované detaily hardware
  - Poskytuje uživateli „virtuální stroj“, který má se snáze ovládá a programuje
- OS jako správce systémových prostředků
  - Každý program dostává prostředky v čase
  - Každý program dostává potřebný prostor na potřebných prostředcích



# Co je pro nás operační systém?

V této přednášce budeme brát operační systém jako **jádro operačního systému**

- ostatní (tzv. systémové) programy lze chápat jako nadstavbu jádra
- GUI – Windows je grafická nadstavba systémových programů








# Různorodost OS

- OS „střediskových“ (mainframe) počítačů – dnes již historický pojem
- OS superpočítačů (3,120,000 jader, 54,902TFlops, 17,8MW příkon)
- OS datových a síťových serverů
- OS osobních počítačů a pracovních stanic
- OS reálného času (Real-time OS – řízení letadel, vlaků, raket, družic, apod.)
- OS přenosných zařízení – telefony, tablety
- Vestavěné OS (tiskárna, pračka, telefon, ...)
- OS čipových karet (smart card OS)
- ... a mnoho dalších specializovaných systémů

# OS pro superpočítače

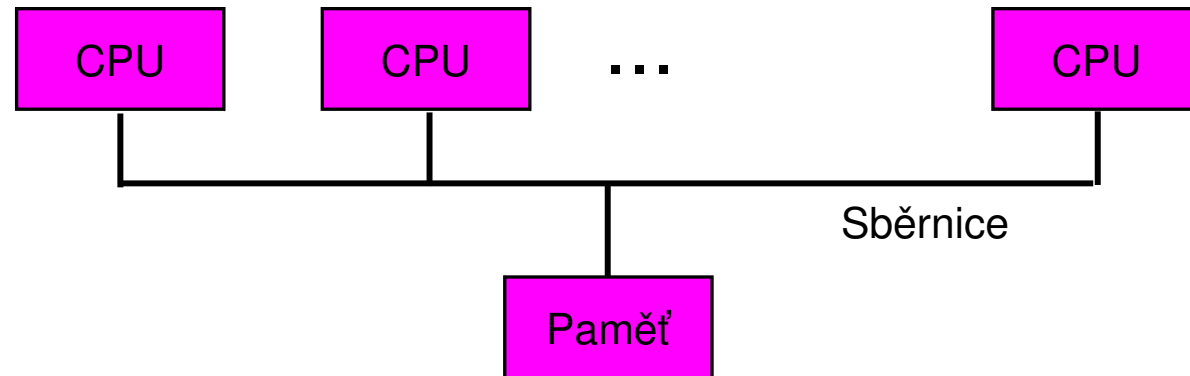
- Počítače pro výpočty velmi složitých simulací a náročných matematických výpočtů
- Zakázky (jobs) se seskupují do dávek (batch) pro nejlepší využití pronajatého strojového času
- Rezidentní program – monitor – předává řízení mezi zakázkami
- Skutečné paralelní spuštění až 3000000 paralelních programů, většinou nepoužívá simulaci paralelismu - multitasking by pouze zdržoval

# Osobní počítač – Personal Computer PC

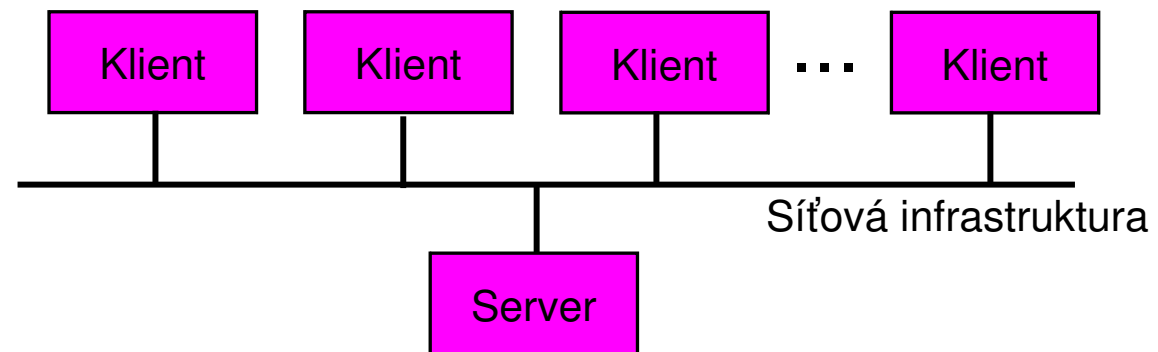
- Typicky orientované na jednoho uživatele
  - v současné době ale vesměs s multiprogramováním (multitaskingem)
- Typizované I/O vybavení
  - klávesnice, myš, obrazovka, disk, USB, malá tiskárna, komunikační rozhraní
- Upřednostňovaným cílem je uživatelské pohodlí
  - minimum ochrany – hlavní roli hraje odpovědnost uživatele
  - často se nevyužívají ochranné vlastnosti CPU
- OS PC často adoptují technologie vyvinuté pro OS větších počítačů
- Mnohdy lze provozovat různé typy operačních systémů
  - M\$ Windows , UNIXy  , Linux , OS X, Android  apod.

# Paralelní a distribuované systémy

Těsně vázaný  
multiprocessorový  
systém



Distribuovaný  
systém typu  
klient-server



# Paralelní systémy

- Zvyšují propustnost a spolehlivost při rozumných nákladech na výpočetní systém
- Multiprocessorové systémy
  - většina současných PC s procesorem s více jádry
  - systémy s více procesory vzájemně komunikujícími vnitřními prostředky jednoho výpočetního systému (např. společnou sběrnici)
- Symetrický multiprocessing (SMP)
  - podporován většinou soudobých OS
  - současně může běžet více procesů na různých CPU/jádrech
  - kterýkoliv proces (i jádro OS) může běžet na kterémkoliv procesoru
- Nesymetrický multiprocessing
  - každý procesor má přidělený specifický úkol
  - hlavní (master) procesor plánuje a přiděluje práci podřízeným (slave) procesorům

# Distribuované systémy

- Rozdělení výpočtů mezi více počítačů propojených sítí
  - lze sdílet zátěž (load-sharing), výpočty se tím zrychlují i za cenu vyšší režie spojené s komunikací
  - zvyšuje se spolehlivost a komunikační schopnosti
  - každý samostatný procesor má svoji vlastní lokální paměť
  - vzájemně se komunikuje pomocí přenosových spojů (sítě) mechanismem výměny zpráv
- Vynucují si použití vhodné síťové infrastruktury
  - LAN, Local Area Networks
  - WAN, Wide Area Networks
- Klasifikace
  - asymetrické distribuované systémy – klient-server
  - symetrické distribuované systémy – peer-to-peer
- OS:
  - Distribuovaný OS vs. síťový OS

# Více úloh současně - Multitasking

- Zdánlivé spuštění více procesů současně je nejčastěji implementováno metodou sdílení času tzv. Time-Sharing Systems (TSS)
- Multitasking vznikl jako nástroj pro efektivní řešení dávkového zpracování
- TSS rozšiřuje plánovací pravidla
  - o rychlé (spravedlivé, cyklické ) přepínání mezi procesy řešícími zakázky interaktivních uživatelů
- Podpora on-line komunikace mezi uživatelem a OS
  - původně v konfiguraci počítač – terminál
  - v současnosti v síťovém prostředí
- Systém je uživatelům dostupný on-line jak pro zpřístupňování dat tak i programů



# Systemy reálného času - RT

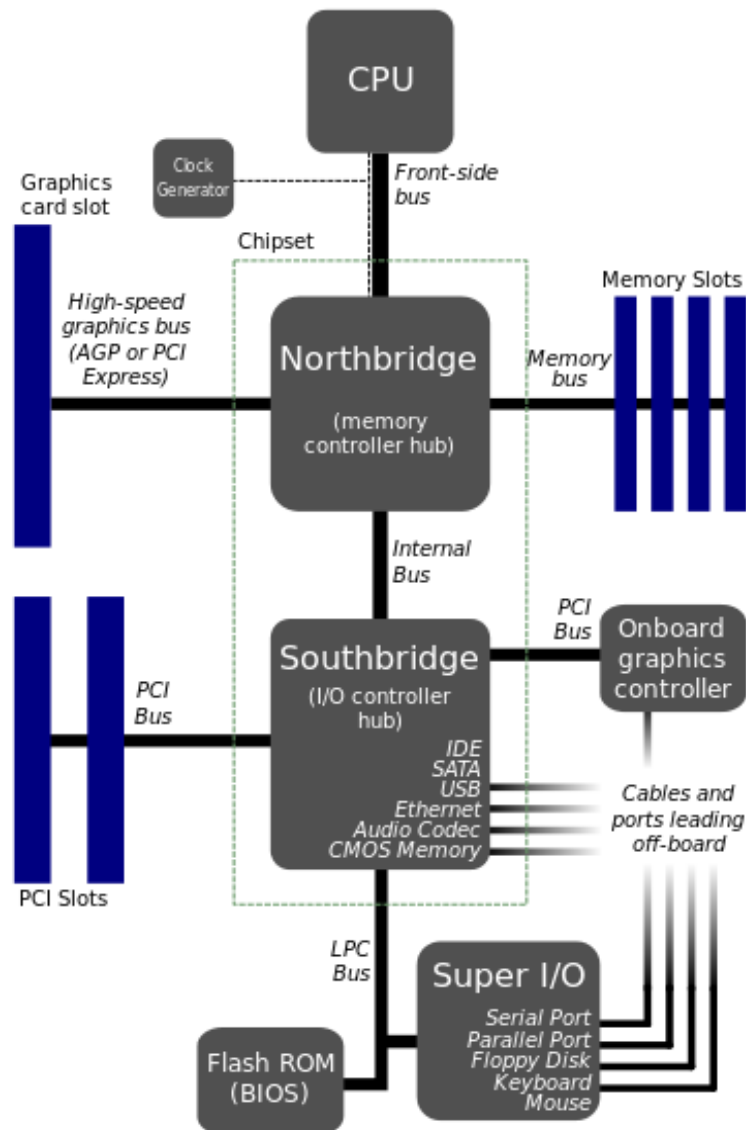
- Nejčastěji řídicí zařízení v dedikovaných (vestavěných) aplikacích:
  - vědecký přístroj, diagnostický zobrazovací systém, systém řízení průmyslového procesu, monitorovací systémy
  - obvykle dobře definované pevné časové limity
  - někdy také subsystém univerzálního OS
- Klasifikace:
  - striktní RT systémy – Hard real-time systems
    - omezená nebo žádná vnější paměť, data se pamatují krátkodobě v RAM paměti
    - protipól univerzálních OS nepodporují striktní RT systémy
    - plánování musí respektovat požadavek ukončení kritického úkolu v rámci požadovaného časového intervalu
  - tolerantní RT systémy – Soft real-time systems
    - použití např. v průmyslovém řízení, v robotice
    - použitelné v aplikacích požadujících dostupnost některých vlastností obecných OS (multimedia, virtual reality, video-on-demand)
    - kritické úkoly mají přednost „před méně šťastnými“

# Přenosné systémy

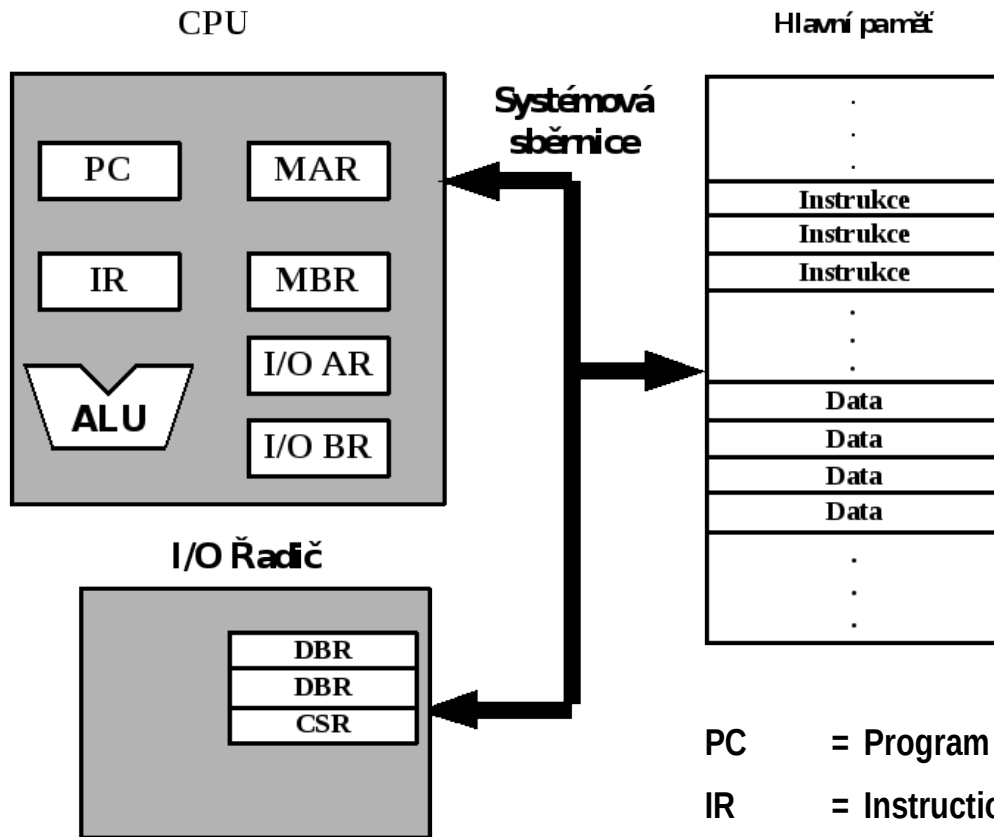
- Handheld Systems, mobilní telefony, tablety
- Charakteristiky:
  - požadavek energetické úspornosti => pomalé procesory
  - speciální obvody na rychlé a efektivní zpracování dat, např. dekódování videa
  - relativně omezená kapacita paměti
  - zpravidla menší display
  - potřeba multiprogramování, avšak obvykle bez sdílení času, výstup na obrazovku pouze jedné aplikace
- Mobilní telefony
  - Navíc podpora síťových komunikačních protokolů
  - Softwarové modemy pro datové přenosy

# OS osobního počítače

- Základem počítače je procesor – CPU
- Procesor je připojen sběrnicemi k ostatním periferiím počítače – paměti, grafickému výstupu, disku, klávesnici, myši, síťovému rozhraní, atd.
- Činnost sběrnice řídí arbiter sběrnice



# Procesor - CPU



Základní vlastnosti:

- šířka datové a adresové sběrnice
- počet vnitřních registrů
- rychlost řídicího signálu – hodiny
- instrukční sada

PC	= Program Counter	= Čítač instrukcí
IR	= Instruction Register	= Registr instrukcí
MAR	= Memory Address Register	= Adresní registr paměti
MBR	= Memory Buffer Register	= Datový vyrovnávací registr paměti
I/O AR	= I/O Address Register	= Adresní registr I/O
I/O BR	= I/O Buffer Register	= Datový vyrovnávací registr I/O
DBR	= Data Buffer Register	= Datový vyrovnávací registr řadiče
CSR	= Control & Status Register	= Řídicí a stavový registr na řadiči

# Registry procesoru

## Řídicí a stavové registry

- PC – program counter – adresa zpracovávané instrukce
- IR – instruction registr – kód zpracovávané instrukce
- PSW – Program status word – stav procesoru  
povoleno/zakázáno přerušení, system/user mód, výsledek operace – přetečení, podtečení, rovnost 0, apod.

## Uživatelské registry

- programově dostupné registry pro ukládání hodnot programu
- obsahují data, adresy, podmínkové kódy
- SP – stack pointer - ukazatel zásobníku, pro ukládání lokálních proměnných a návratových adres funkcí

# Režimy práce procesoru

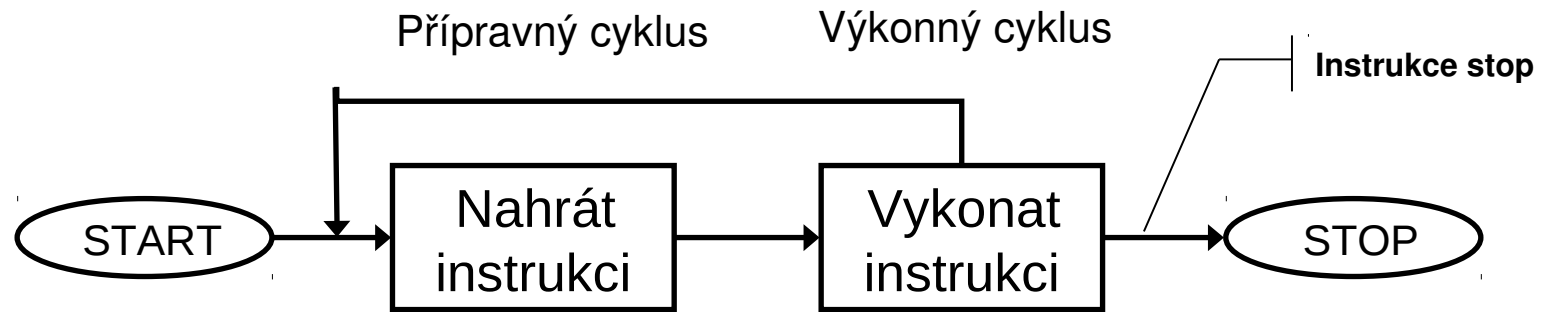
Dva režimy práce procesoru - základ hardwarových ochran

- Systémový = privilegovaný režim
  - procesor může vše, čeho je schopen
- Uživatelský = aplikační (ochranný) režim
  - privilegované operace jsou zakázány
- Privilegované operace
  - ovlivnění stavu celého systému (halt, reset, Interrupt Enable/Disable, modifikace PSW, modifikace registrů MMU )
  - instrukce pro vstup/výstup (in, out)
- Okamžitě platný režim je zachycen v PSW (S-bit)

Přechody mezi režimy

- Po zapnutí stroje systémový režim
- Přechod do uživatelského – modifikace PSW
- Přechod do systémového – pouze přerušení vč. programového

# Pracovní krok procesoru



Procesor pracuje v krocích. Jeden krok obsahuje fáze:

- Přípravná fáze (fetch cycle)
  - nahrává do procesoru instrukci podle PC a umístí její kód do IR
  - na jejím konci se (zpravidla) inkrementuje PC
- Výkonná fáze (execute cycle)
  - vlastní provedení instrukce
  - může se dále obracet (i několikrát) k paměti

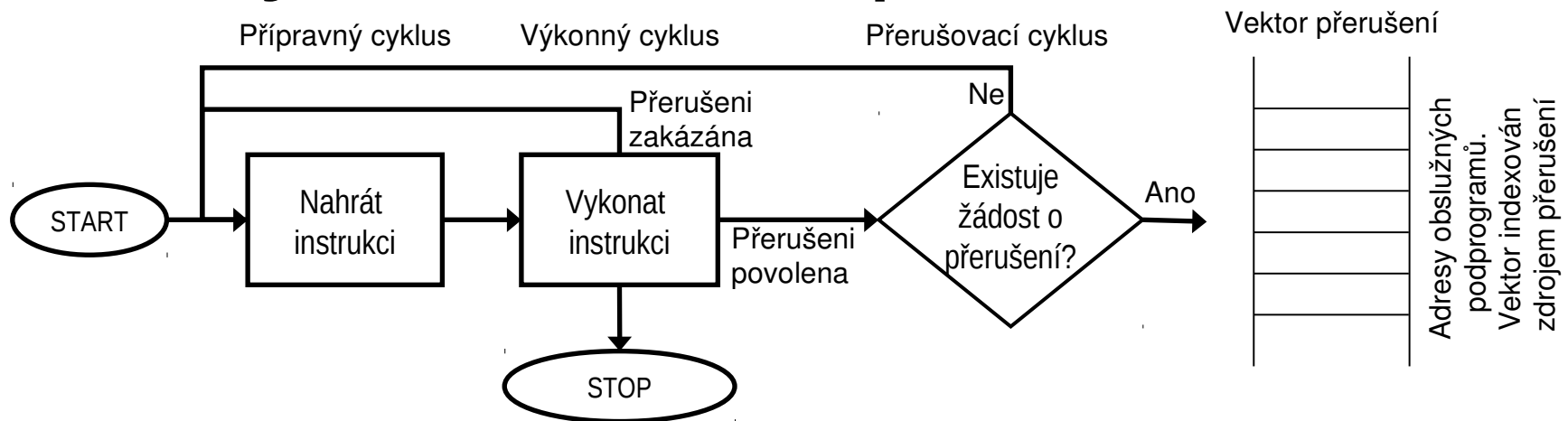
```
loop:  FETCH; /* ((PC)) → IR */  
      Increment(PC);  
      EXECUTE; /* provede operaci dle (IR) */  
end loop
```

# Přerušeni

- Přerušeni normální posloupnosti provádění instrukcí
  - cílem je zlepšení účinnosti práce systému
  - je potřeba provést jinou posloupnost příkazů jako reakci na nějakou „neobvyklou“ externí událost
  - přerušující událost způsobí, že se pozastaví běh procesu v CPU takovým způsobem, aby ho bylo možné později znovu obnovit, aniž by to přerušovaný proces „poznal“
- Souběh I/O operace
  - přerušeni umožní, aby CPU prováděla jiné akce než instrukce programu čekajícího na konec I/O operace
  - činnost CPU se později přeruší iniciativou „I/O modulu“
  - CPU předá řízení na obslužnou rutinu přerušeni (Interrupt Service Routine) – standardní součást OS
- CPU testuje nutnost věnovat se obsluze přerušeni alespoň po dokončení každé instrukci
  - existují výjimky (např. „blokové instrukce“ Intel)



# Cyklus CPU s přerušením



```
INTF=False; /* je žádost o přerušování ? */
```

```
loop: FETCH;
```

```
Increment(PC);
```

```
EXECUTE;
```

```
if povoleno přerušování && INTF then
```

```
    Ulož PSW na zásobník;
```

```
    Ulož PC na zásobník;
```

```
    PSW nastav System mode a Interrupt disabled;
```

```
    PC = vektoru přerušování podle čísla přerušování
```

```
end loop
```

# Přerušení a jejich druhy

Přerušení je speciálním případem výjimečné situace.

Synchronní přerušení (s během programu)

- Programové (naprogramované) - speciální instrukce (INT, TRAP)
- Generované kontrolními obvody počítače:
  - dělení nulou, pokus o vykonání nelegální či neznámé instrukce
  - neoprávněný pokus o přístup k paměťové lokaci (narušení ochrany paměti, virtuální paměť)

Asynchronní (přicházející zvenčí – klasické přerušení)

- I/O, časovač, hardwarové problémy (např. výpadek napájení ...)

Kdy se na výjimečné situace reaguje?

- Standardní přerušení: po dokončení instrukce během níž vznikl požadavek
- Výjimka vysoké úrovně: během provádění instrukce (po dokončení některé fáze provádění instrukce) – instrukci nelze dokončit – neznámá instrukce, dělení nulou
- Kritická výjimka: nelze dokončit ani cyklus přenosu dat a je nutno reagovat okamžitě – narušení ochrany paměti

# Obsluha přerušení

Žádost se vyhodnotí na přípustnost (priority přerušení)

Procesor přejde do zvláštního cyklu

- PSW se uloží na zásobník (Výsledek aritmetických operací se mění téměř každou instrukcí).
- Na zásobník se uloží i čítač instrukcí PC (návrátová hodnota z přerušení).
- Do PSW se vygeneruje nové stavové slovo s nastaveným S-bitem. Nyní je CPU v privilegovaném režimu
- PC se nahradí hodnotou z vektoru přerušení - skok na obsluhu přerušení

Procesor přechází do normálního režimu práce a zpracovává obslužnou rutinu přerušení

- Obslužná rutina musí být transparentní, tj. programově se musí uložit všechny registry CPU, které obslužná rutina použije, a před návratem z přerušení se opět vše musí obnovit tak, aby přerušená posloupnost instrukcí nepoznala, že byla přerušena.
- Obslužnou rutinu končí instrukce „návrat z přerušení“ (IRET, RTE) mající opačný efekt: z vrcholu zásobníku vezme položky, které umístí zpět do PC a PSW

# Vícenásobná přerušeni

## Sekvenční zpracování

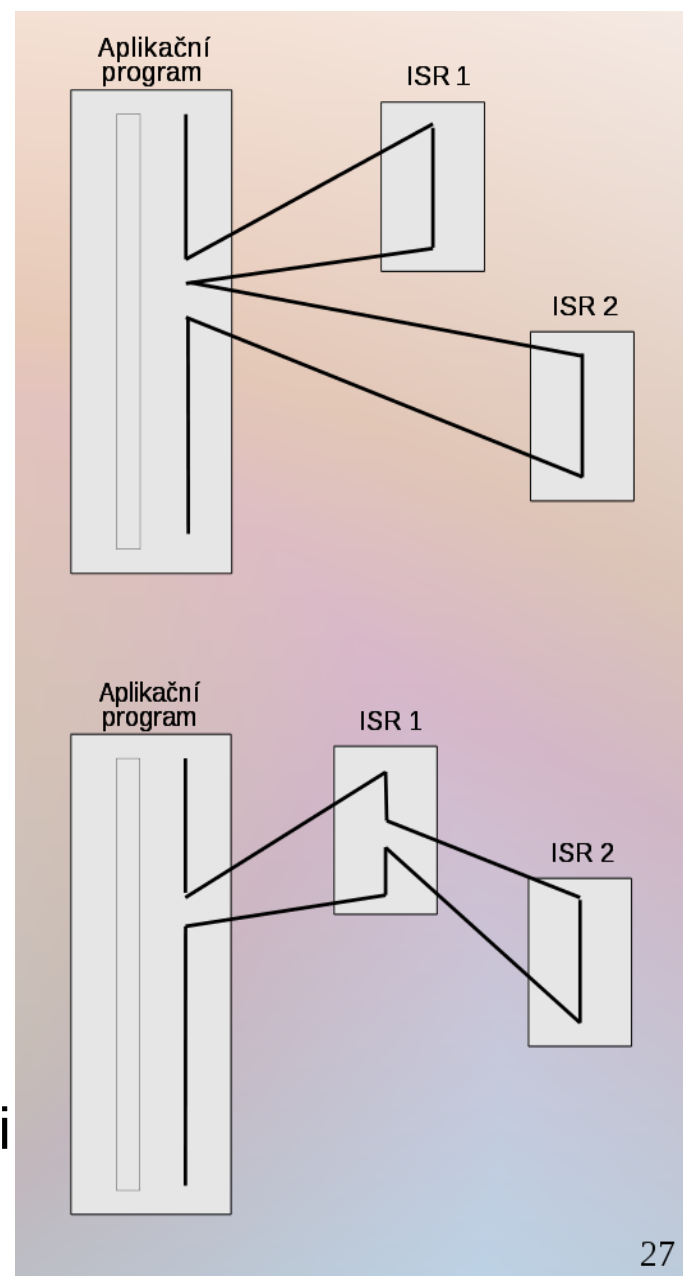
- během obsluhy jednoho přerušeni se další požadavky nepřijímají (pozdržují se)
- jednoduché, ale nevhodné pro časově kritické akce

## Vnořené zpracování

- prioritní mechanismus
- přijímají se přerušeni s prioritou striktně vyšší, než je priorita obsluhovaného přerušeni

## Odložené zpracování

- V přerušeni se provede pouze nejnutnější obsluha zařízení, zbytek se provede v rámci OS
- Neblokují se zbytečně další přerušeni



# Obsluha I/O zařízení

I/O operace:

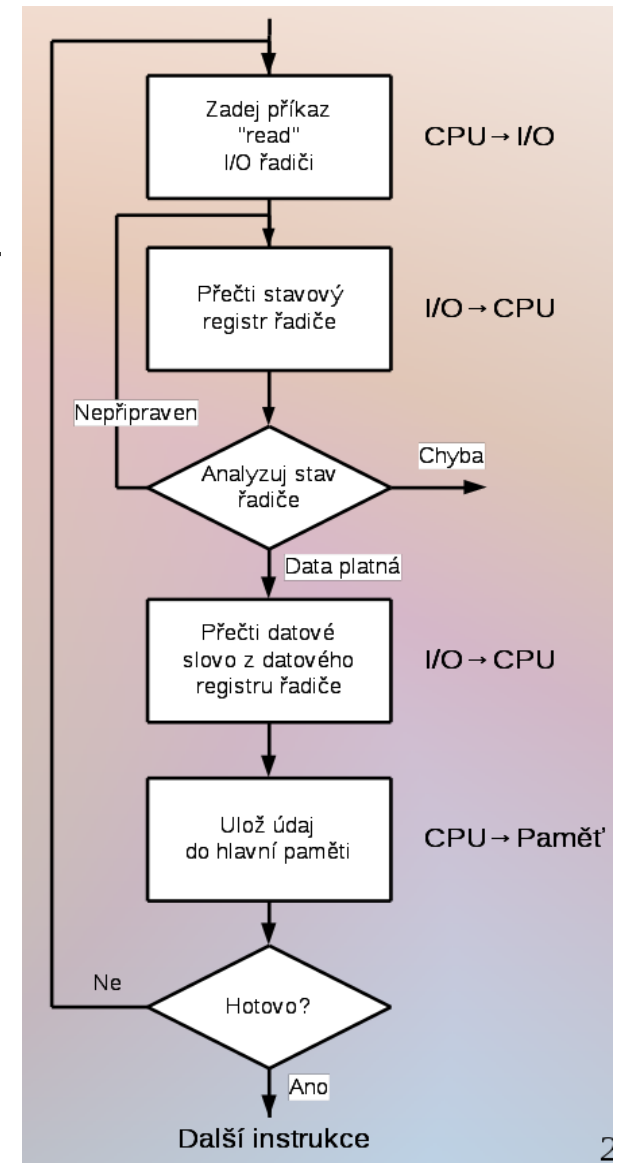
- Zpravidla přenos sekvence údajů přes sběrnici
- Přenos dat z I/O zařízení do CPU – vstup,
- Přenos dat z CPU do I/O – výstup

Dva způsoby obsluhy

- S aktivním čekáním (busy waiting)
  - v systémech bez řízení IO pomocí OS
  - žádné souběžné zpracovávání I/O, nedořešený zůstává nejvýše jeden I/O požadavek
  - program testuje konec IO operace opakovanými dotazy na příslušný stavový registr IO zařízení
- S přerušením a OS řízeným souběžným prováděním
  - v systémech s řízením IO pomocí OS
  - souběžné zpracovávání I/O paralelně s během programu(ů)
  - I/O operaci zahajuje OS na žádost z uživatelské ho procesu
  - uživatelský proces čeká na dokončení I/O operace – synchronní řešení
  - uživatelský proces nečeká na dokončení I/O operace – asynchronní řešení I/O, může běžet souběžně více I/O operací

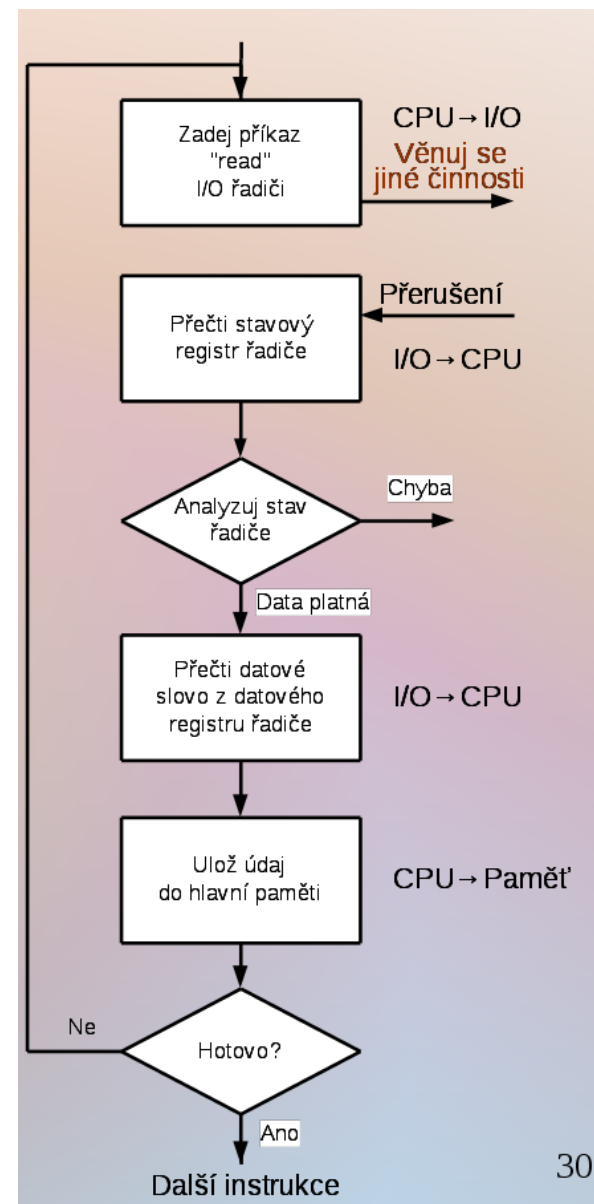
# I/O a aktivním čekáním

- CPU zahajuje elementární přenos údajů a v „dotazovací smyčce“ čeká na připravenost dat
- programově velmi jednoduché
- velmi neefektivní, ale velmi rychlá reakce
- až na zcela výjimečné případy
- použitelné jen v primitivních systémech bez multiprogramování, nebo jako první část obsluhy I/O zařízení



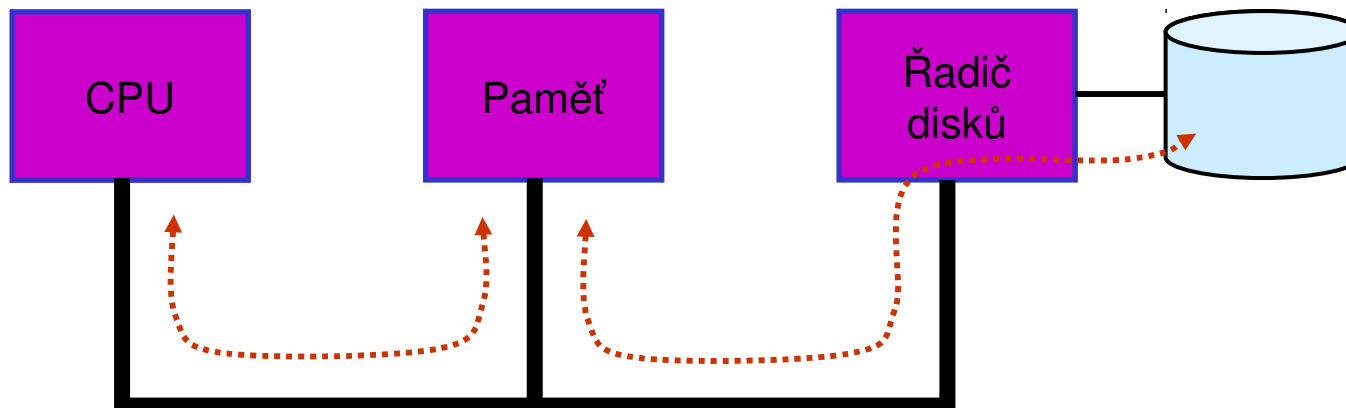
# I/O obsluha s přerušením

- CPU inicializuje elementární přenos a věnuje se jiné činnosti
- Když je údaj připraven, I/O zařízení vyvolá přerušeni
- Obslužná rutina přenese data mezi zařízením a pamětí
- Pružné – data lze při přenosu upravovat
- Relativně pomalé, účast CPU, řízeno programem
- Jen pro zařízení schopná práce v režimu start-stop
  - Zařízení schopná ze své fyzikální podstaty pozastavit přenos dat kdykoliv a na libovolně dlouhou dobu beze ztrát



# Přímý přístup do paměti - DMA

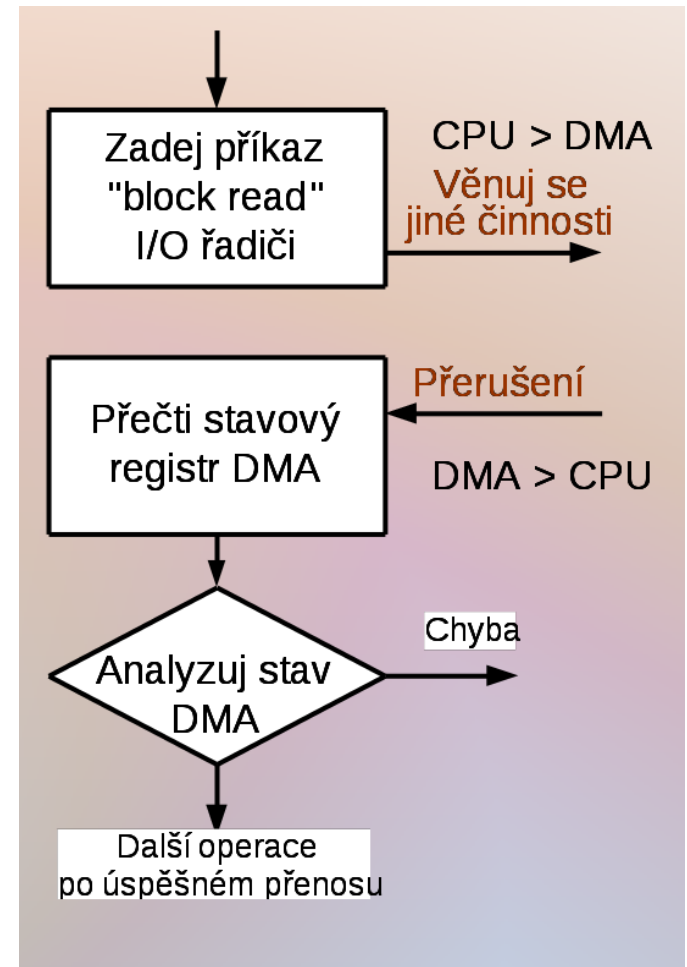
- Určeno pro blokové přenosy dat vysokou rychlostí
- I/O přenosy se uskutečňují bez přímé účasti procesoru mezi periferním zařízením a pamětí
- Procesor dovolí I/O modulu přímo číst z nebo zapisovat do operační paměti – kradení cyklů (cycle stealing)
- Procesor zadá jen velikost a umístění bloku v paměti a směr přenosu
- Přerušení se generuje až po dokončení přenosu bloku dat





# I/O obsluha s DMA

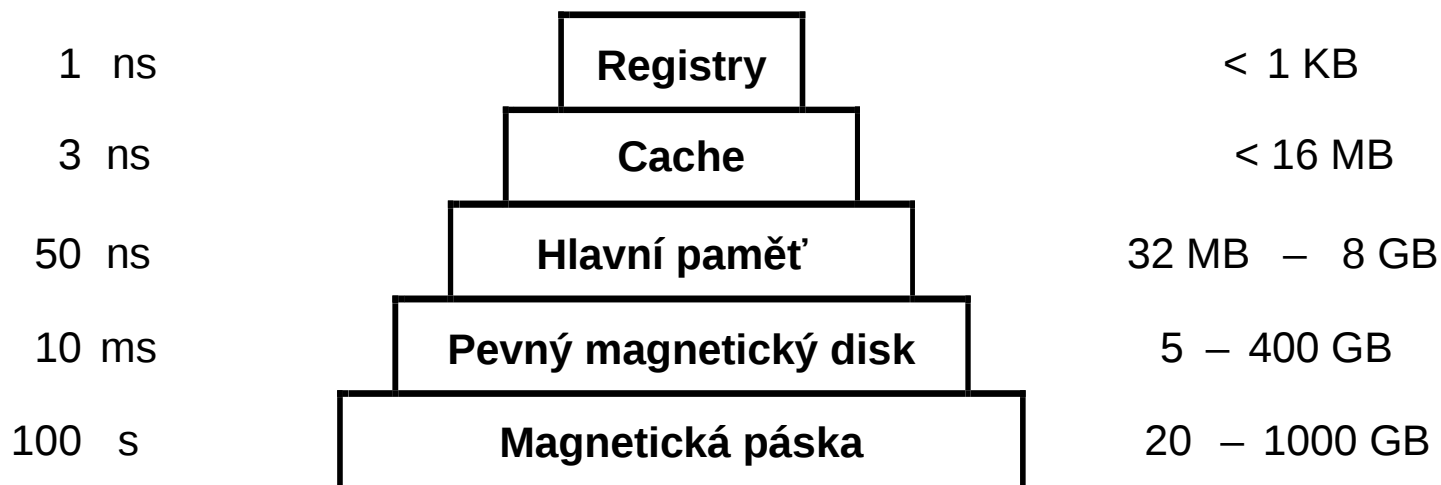
- CPU zadá parametry přenosu DMA jednotce
- Přenos probíhá autonomně bez účasti CPU
- DMA vyvolá přerušení po ukončení přenosu bloku (nebo při chybě)
- Obslužná rutina pouze testuje úspěšnost přenosu a informuje OS, že přenos skončil



# Druhy pamětí

- Hierarchie pamětí z pohledu rychlosti a kapacity
  - uvedená čísla představují pouze hrubá přiblížení
  - směrem dolů klesá rychlost i „cena za 1 bit“
- Typy prvků používaných v hlavní paměti
  - RAM, ROM, EEPROM, CMOS-RAM

*Typická přístupová doba*



# Hierarchie pamětí

Úroveň	1	2	3	4
Označení	Registry CPU	cache	Hlavní paměť	disk
Typická velikost	$\leq 1\text{KB}$	$\leq 16\text{MB}$	$\sim 16\text{ GB}$	$> 100\text{ GB}$
Technologie	Uvnitř CPU (CMOS)	CMOS SRAM	CMOS SRAM	Magnetický disk / SSD
Přístupová doba	$\sim 0,5\text{ns}$	$\sim 1\text{-}25\text{ns}$	$\sim 80\text{-}500\text{ns}$	$\sim 5\text{ms}$
Správce	program	HW	operační systém	operační systém
Simuluje	hlavní paměť	hlavní paměť	disk	

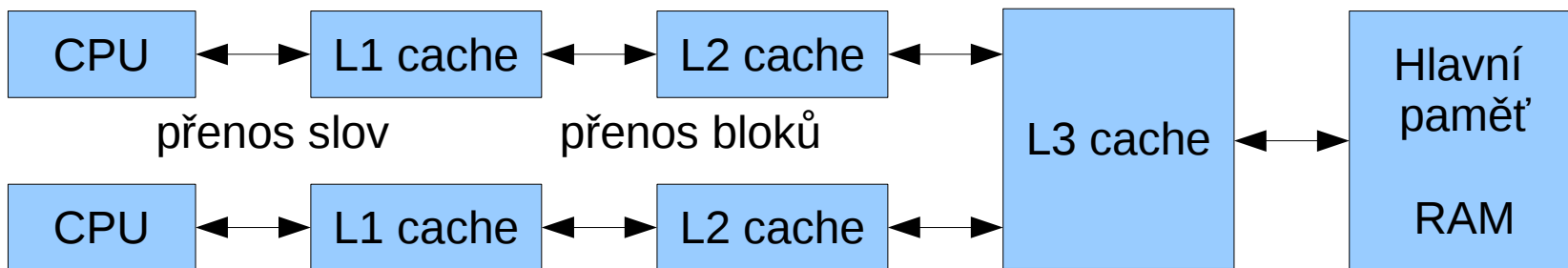
# Caching, cache paměti

Caching je princip používaný v OS velmi často

- části obsahu pomalejší paměti s vyšší kapacitou jsou podle potřeby dočasně kopírovány do paměti rychlejší

Mezipaměť ležící mezi CPU a hlavní pamětí

- Transparentní pro operační systém i pro programátora
- Je rychlejší než operační (hlavní) paměť
- Mikroprogramem řízené kopírování informací z hlavní paměti do cache paměti po blocích
- Princip časové a prostorové lokality běžných programů
- Problém udržení konzistence více kopií těchto dat v multiprocessorových systémech



# Cache

## Velikost cache

- čím větší, tím častěji se najdou požadovaná data v cache, ale také roste cena

## Velikost přenosového bloku – kompromis:

- velké bloky = dlouhé přenosy
- malé bloky = časté přenosy

## Mapovací funkce

- kam přijde blok do cache

## Nahrazovací algoritmus:

- určuje, který blok v cache bude nahrazen
- Least-Recently-Used (LRU) algoritmus

## Analogie

- hardwarově realizované principy původně vyvinuté pro virtuální paměť

# Bezpečnostní mechanismy v OS

## Základní opatření

- Dva režimy práce procesoru(ů)
- Vstup a výstup: Povinné a uživatelským režimem vynucené volání služeb OS - I/O instrukce jsou privilegované
- Uživatelský program nikdy nesmí získat možnost práce v privilegovaném režimu
  - Např. nesmí mít možnost zapsat do PSW a změnit tak režim práce CPU (S-bit v PSW) nebo modifikovat vektor přerušení

## Ochrana dostupnosti CPU

- Prevence před převzetím vlády jednoho aplikačního programu nad CPU
- Řešení: časovač (timer)
  - V pravidelných (privilegovaně programovatelných) intervalech vyvolává přerušení, a tak je aktivováno jádro OS
  - Mnohdy realizován jako „periferní zařízení“
  - O přerušení od časovače se opírají mechanismy plánování procesoru(ů)

# Bezpečnostní mechanismy v OS

## Funkcionalita pro správu paměti

- Systém musí být schopný přidělovat paměť různým zakázkám a ze zakázek odvozeným procesům dynamicky přidělovat paměť
- Dvojí pohled na paměť
  - z hlediska její fyzické konstrukce a šířky fyzických adresovacích sběrnic – fyzický adresní prostor, FAP
  - z hlediska konstrukce adresy ve strojovém jazyku – logický adresní prostor, LAP
- Ochrana oblastí paměti před neautorizovaným přístupem

# Bezpečnostní mechanismy v OS

## Mechanismus přerušení

- předávání řízení mezi uživatelským programem a systémem → implementace reakcí na asynchronní události
- OS je systém řízený přerušeními

## Plánování práce CPU

- reaguje se na generátor časových značek (timer) – po uplynutí daného intervalu generuje přerušeni
- ochrana proti trvalému obsazení CPU uživatelským procesem (záměrně, chybou, ...)
- OS musí být schopen volit mezi různými výpočetními procesy připravenými k činnosti



To je dnes vše.

Otázky?