# PRINCIPY POČÍTAČŮ

Historický vývoj, předpoklady fungování, binární logika, modulace signálu.

## HISTORICKÝ VÝVOJ

* **Předchůdci**
  + Abbakus (podobný princip jako počítadlo)
  + mechanické kalkulátory (da Vinci, Schickard, Pascal, von Liebnitz, de Colmar)
  + Babbageův analytický stroj (počítač poháněný párou)
  + logaritmické pravítka, elektromechanické kalkulátory
* **I. Generace**
  + vývoj začal během 2. Světové války
  + neexistoval jednotný software – každý počítač – vlastní program zapsaný ve strojovém kódu
  + elektronky → velký rozměr počítačů
  + I/O – mechanická záznamová média – páska, děrné štítky
  + náročné na údržbu
  + většina dat uložena na přenosných médiích, hlavní paměť – pevné nevyměnitelné disky (40 až 50 kb)
  + počítač mohla využívat pouze jedna osoba
  + výkon 103-104 oper/sec
* **II. Generace**
  + 3 nejnaléhavější problémy – zvýšení rychlosti provádění operací, zvýšení kapacity paměti a zvýšení rychlosti přenosu dat na vstupu a výstupu
  + vyřešeno s příchodem polovodičů (1948, tranzistor, dioda)
  + od roku 1956 používány v počítačích
  + dochází ke zmenšování počítačů
  + velký vývoj pamětí – magnetická jádra, magnetické pásky, první diskové paměti
  + 1. univerzální vyšší programovací jazyky
  + výkon kolem 105 oper/sec
* **III. Generace**
  + tranzistory – velké množství tepla – časté poškození
  + 1958 - Jack St. Clair Kilby – integrovaný obvod
  + velké zvýšení výkonu, zmenšení rozměrů
  + konec děrných štítků, pásek, magnetických jader, nástup paměťových disků
  + zlepšení výstupu – LED diody, obrazovky
  + vylepšení programovacích jazyků
  + 1. osobní počítače a sálové superpočítače
  + výkon až 106 oper/sec
* **IV. Generace**
  + 1968 – vylepšení integrovaných obvodů – miniaturizace
  + mikroprocesory – 1971
  + nová paměťová média – FDD, HDD
  + začátek masové výroby osobních počítačů určených pro použití v kanceláři i v rodinách
  + neustálý vývoj nových technologií =>zvyšování výkonu, miniaturizace, snížení pořizovacích nákladů
  + dynamické paměti, BIOS
  + výkon přes 106 oper/sec
  + Intel 4040 (1969, IBM, USA), Intel 8080 (1972, USA, v ČSSR MHB8080A), Sinclair ZX Spectrum (GB, 1982), 80386SX (1988, USA), Pentium (1996, USA)
* **V. Generace**
  + matice mikroprocesorů
  + transputery (TRANZISTOR a COMPUTER) – kompletní počítač na čipu, který tvoří základní stavební jednotkou procesorového systému)
  + optické prvky od roku 1990 (CD, DVD)
  + výkon až 109 oper/sec
  + vstup a výstup povelů v lidské řeči

## Číselné soustavy

* **Desítková soustava** – rozlišení 10 cifer (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
  + Váha cifry je dána pozicí v čísle, 1869 chápeme jako:

1·103 + 8·102 + 6·101 + 9·100

* + V číselné soustavě s obecným základem z lze číslo zapsat:

X = an·zn + an-1·zn-1 + … + a2·z2 + a1·z1 + a0·z0

* + Zapisujeme: an, an-1, … a2, a1, a0
  + Základ z může být libovolný – nutnost použití právě **z** cifer
* **Binární soustava** – pro počítačové systémy vhodnější
  + Rozlišující dva stavy – 0 a 1 představující hodnotu elektrického napětí (0 V nebo > 0 V).
  + Jednotkou informace je bit.

1 bit ~ 1b

1 0 1 0 = 4b ~ tetráda ~ NIBL

0 1 1 1 0 1 0 1 = 8b ~ 1 Byte ~ 1B

* + zápis binárních hodnot: 0101 1100 1001 0111
* **Octalová** (osmičková) **soustava**
  + 3 binární bity = 1 octalová cifra
  + Rozsah 000-111
  + Cifry 0-7 → tedy osm cifer (23 = 8)
* **Hexadecimální** (šestnáctková) **soustava**
  + 4 bity = 1 hexadecimální cifra
  + 16 cifer (24 = 16)
  + 1B = 2 hexadecimální cifry

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 0 0 0 | 0 |  | 1 0 0 0 | 8 |
| 0 0 0 1 | 1 |  | 1 0 0 1 | 9 |
| 0 0 1 0 | 2 |  | 1 0 1 0 | A |
| 0 0 1 1 | 3 |  | 1 0 1 1 | B |
| 0 1 0 0 | 4 |  | 1 1 0 0 | C |
| 0 1 0 1 | 5 |  | 1 1 0 1 | D |
| 0 1 1 0 | 6 |  | 1 1 1 0 | E |
| 0 1 1 1 | 7 |  | 1 1 1 1 | F |

## PŘEVOD MEZI SOUSTAVAMI

* Číslo, které chceme převést, dělíme neustále dvojkou, až dojdeme k nule, přičemž si zapisujeme zbytky po celočíselném dělení. Pokud chceme převést číslo do jiné soustavy, například do šestnáctkové, budeme dělit šestnáctkou. Pokud do šestkové, dělíme šestkou.
* Výsledné číslo jsou zbytky řazené od konce, tzn.:
  + (120)10 = (1111000)2

120 : 2 = 60 → 0 (zbytek)

60 : 2 = 30 → 0

30 : 2 = 15 → 0

15 : 2 = 7 → 1

7 : 2 = 3 → 1

3 : 2 = 1 → 1

1 : 2 = 0 → 1

* + (130)10 = (244)7

130 : 7 = 18 → 4 (zbytek)

18 : 7 = 2 → 4

2 : 7 = 0 → 2

* Obrácený postup, resp. zkouška:

1·26 + 1·25 + 1·24 + 1·23 + 0·22 + 0·21 + 0·20 = 64 + 32 + 16 + 8 = 120

2·72 + 4·71 + 4·70 = 98 + 28 + 4 = 130

## biNÁRNÍ LOGIKA

* Zabývá se logickými operacemi ve dvojkové soustavě
* V praxi jsou tyto operace vykonávány na hradlech, popř. klopné obvody s několika stabilními a nestabilními stavy
* Dnes spíše integrované obvody, které nabízejí více funkcí
* **Logický součet** – **OR** (X V Y = Z; X + Y = Z)
  + Pokud platí alespoň jeden ze vstupů X a Y, pak platí výstup Z

|  |  |  |
| --- | --- | --- |
| X | Y | Z |
| 0 | 0 | **0** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **1** |

* **Logický součin – AND** (X Λ Y = Z; X \* Y = Z)
  + Pokud platí oba vstupy X a Y, pak platí výstup Z

|  |  |  |
| --- | --- | --- |
| X | Y | Z |
| 0 | 0 | **0** |
| 0 | 1 | **0** |
| 1 | 0 | **0** |
| 1 | 1 | **1** |

* **Negace** (⌐X = Y; X̅ = Y)
  + Pokud platí vstup X, pak výstup Y neplatí

|  |  |
| --- | --- |
| X | Y |
| 0 | **1** |
| 1 | **0** |

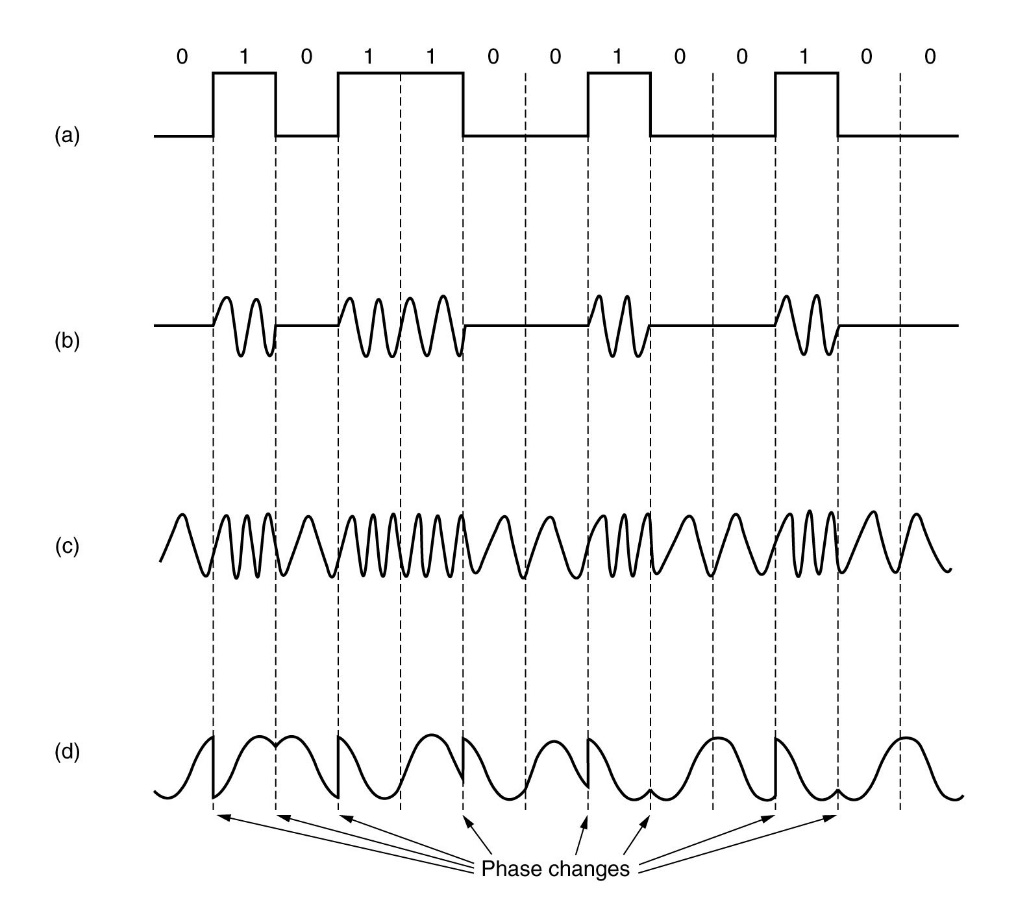
* **Exkluzivní logický součet – XOR** (X xor Y = Z)
  + Výstupem je exkluzivní logický součet vstupů. Výstup je 1 jen tehdy, pokud se hodnoty vstupů liší.

|  |  |  |
| --- | --- | --- |
| X | Y | Z |
| 0 | 0 | **0** |
| 1 | 0 | **1** |
| 0 | 1 | **1** |
| 1 | 1 | **0** |

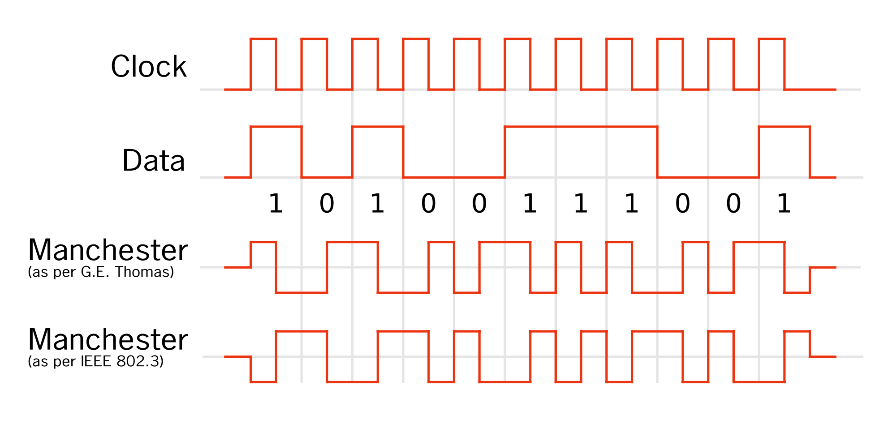
* **Negovaný logický součet** **– NOR** (X+Y= Z)
  + Pokud neplatí ani jeden ze vstupů X a Y, pak platí výstup Z
  + Negace výsledku OR
* **Negovaný logický součin – NAND** (X\*Y= Z)
  + Pokud zároveň neplatí vstupy X a Y, pak platí výstup Z
  + Negace výsledku AND
* **Axiomy**
  + komutativita (nezávislost na pořadí operandů): X and Y = Y and X
  + distributivita (možnost různé distribuce funkce přes jinou): (X and Y) and Z = X and (Y and Z)
  + neutralita: X and 1 = X; X or 0 = X
  + komplementarita: X and ⌐X = 0; X or ⌐X = 1

## MODULACE SIGNÁLU

* Ideálním průběhem analogového signálu je **spojitá hladká křivka**, nejčastěji sinusovka. Ideálním průběhem digitálního signálu jsou pak **obdélníkové pulsy**. Signál je ale vždy více či méně degradován (znehodnocován) "šumem" a "útlumem".
* **Útlum** = při přenosu signálu po přenosovém kanále (mediu) vždy dochází k energetickým ztrátám.
  + Amplituda přenášeného signálu tedy slábne úměrně ke vzdálenosti.
* **Šum =** vznik nežádoucích, slabších či silnějších signálů jiné frekvence, než je přenášený signál.
  + Skládáním těchto signálů s přenášeným dochází k "interferenci", tvar přenášeného signálu se mění.
  + Příčina vznik rušivých signálů – šumů – je závislá na nosném mediu. Například při přenosu signálu elektrickým obvodem – vodičem – může docházet k elektrické magnetické indukci od souběžného elektrického vedení.
  + Rušivé signály můžeme eliminovat vhodnou volbou kabelu (např. kroucená dvoulinka), stíněním a zemněním, apod.
* Zařízení, které provádí modulaci, se nazývá modulátor. Opakem modulace je demodulace, kterou provádí demodulátor.
* Druhy modulace
  + **Amplitudová modulace** (b) – podle změn modulovaného signálu se mění amplituda nosného signálu. Nevýhodou tohoto způsobu je především situace, kdy je modulovaný signál nulový – pak je i nosný signál nulový.
  + **Frekvenční modulace** (c) – podle změn modulovaného signálu se mění frekvence nosného signálu. Nosný signál má stále stejnou amplitudu. Oproti amplitudové modulaci tedy nemůže dojít k záměně výpadku vysílače s nulovým signálem. Velké změny nosného kmitočtu však někdy mohou činit problémy s naladěním přijímače.
  + **Fázová modulace** (d) – podle změn modulovaného signálu se mění fáze (dochází k fázovému posunu) nosného signálu. Eliminuje nevýhody obou předešlých typů, ale je složitá na realizaci.



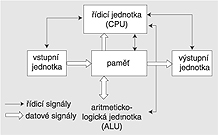
* **Kódování**
  + V různých elektronických obvodech jsou dva možné stavy (0 a 1) reprezentovány různými hodnotami. Např. v TTL logice je "0" reprezentována hodnotou 0V a "1" hodnotou 5V. Přenášíme-li takovýto digitální signál pomocí modulace, tak například amplitudová modulace při přenosu řady signálu s hodnotou "0" by prakticky nepřenášela nic. Proto digitální signál kódujeme, tj. nastavujeme mu časování, neboli „rytmus“ shodný na obou stranách, který umožňuje rozpoznat jednotlivé bity.



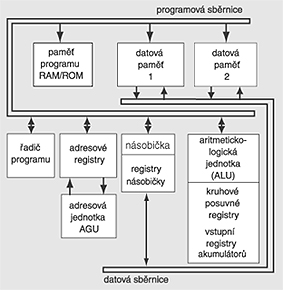
# Architektura počítače

Von neumannovo a harwardské schéma, flynnova taxonomie, základní deska, procesor, mikroarchitektura procesoru, paměti, sběrnice, řadič, přídavné karty, ovladače.

## Von Neumannovo schéma

* Navrženo roku 1945 americkým matematikem **Johnem von Neumannem** jako model samočinného počítače
* Tento model s jistými výjimkami zůstal zachován dodnes
* **Podle tohoto schématu se počítač skládá z pěti hlavních modulů**
  + **Operační paměť**
    - slouží k uchování
      * zpracovávaného programu
      * zpracovávaných dat
      * výsledků výpočtu
  + **ALU**
    - *Arithmetic-logic Unit* (aritmetickologická jednotka)
    - jednotka provádějící veškeré aritmetické výpočty a logické operace.
    - obsahuje
      * sčítačky
      * násobičky (pro aritmetické výpočty)
      * komparátory (pro porovnávání)
  + **Řadič**
    - řídící jednotka, která řídí činnost všech částí počítače
    - toto řízení je prováděno pomocí řídících signálů, které jsou zasílány jednotlivým modulům
    - reakce na řídící signály, stavy jednotlivých modulů jsou naopak zasílány zpět řadiči pomocí stavových hlášení
  + **Vstupní zařízení**
    - zařízení určená pro vstup programu a dat.
  + **Výstupní zařízení**
    - zařízení určená pro výstup výsledků, které program zpracoval
  + Ve Von Neumannově schématu je možné ještě vyznačit dva další moduly vzniklé spojením předcházejících modulů:
    - **Procesor**
      * Řadič + ALU
    - **CPU**
      * *Central Processing Unit* (centrální procesorová jednotka):
      * Procesor + Operační paměť
* **Von Neumannova architektura**
  + počítač (mikroprocesor) se skládá z pěti funkčních bloků – paměti, řadiče, aritmeticko-logické jednotky (ALU) a vstupních a výstupních jednotek
  + struktura počítače je nezávislá na typu řešení úlohy a je řízena obsahem paměti
  + instrukce a operandy jsou uloženy v téže paměti,
  + paměť je rozdělena do buněk stejné velikosti, jejichž pořadová čísla se používají jako adresy
  + program je tvořen posloupností elementárních příkazů (instrukcí), v nichž zpravidla není obsažena hodnota operandu (uvádí se pouze jeho adresa), takže program se při změně dat nemění
  + instrukce se provádějí jednotlivě v pořadí, v němž jsou zapsány do paměti
  + změna provádění instrukcí se vyvolá instrukcí podmíněného a nepodmíněného skoku
  + pro reprezentaci instrukcí a čísel (operandů, výsledků, adres apod.) se používají dvojkové signály a dvojková číselná soustava.
* Tato koncepce ovlivnila výrobu počítačů a pak i mikroprocesorů až do osmdesátých let minulého století, kdy se objevil první signálový procesor s harvardskou architekturou
* **Princip činnosti počítače podle von Neumannova schématu**
  1. Do operační paměti se pomocí vstupních zařízení přes ALU umístí program, který bude provádět výpočet
  2. Stejným způsobem se do operační paměti umístí data, která bude program zpracovávat
  3. Proběhne vlastní výpočet, jehož jednotlivé kroky provádí ALU.
     + Tato jednotka je v průběhu výpočtu spolu s ostatními moduly řízena řadičem počítače
     + Mezivýsledky výpočtu jsou ukládány do operační paměti.
  4. Po skončení výpočtu jsou výsledky poslány přes ALU na výstupní zařízení.
* **Základní odlišnosti dnešních počítačů od von Neumannova schématu**
  + Podle von Neumannova schématu počítač pracuje vždy nad jedním programem
    - Toto vede k velmi špatnému využití strojového času  
      Je tedy obvyklé, že počítač zpracovává paralelně více programů zároveň - tzv. **multitasking**
  + Počítač může disponovat i více než jedním procesorem – tzv. **multiprocessing**
  + Počítač podle von Neumannova schématu pracoval pouze v tzv. diskrétním režimu
  + Existují vstupní / výstupní zařízení, která umožňují jak vstup, tak výstup dat (programu)
  + Program se do paměti nemusí zavést celý, ale je možné zavést pouze jeho část a ostatní části zavádět až v případě potřeby
* **Diskrétní režim** 
  + **Protected mode - chráněný režim, též ochranný režim**
  + Provozní režim procesorů Intel 80x86 a kompatibilních, při kterém je k dispozici
    - větší paměťový prostor
    - možnosti multitaskingu
    - možnost využití virtuální paměti (**program může zasahovat pouze do své přidělené paměti)**
    - v chráněném režimu pracuje automaticky operační systém OS/2, Windows NT a většina systémů UNIX
* **Reálný režim (real mode)**
  + Druh paměťového režimu u počítačů založených na procesoru Intel 80x86
  + Je to jediný režim, který přímo podporuje operační systém MS-DOS
  + Hlavními znaky tohoto režimu je, že
    - podporuje pouze jednoúlohové prostředí (tedy neumožňuje pravý multitasking)
    - dokáže přímo zpřístupnit pouze paměť do 1 MB
  + Druhým režimem, který podporují procesory této řady, je chráněný režim
* **Multiprocessing**
  + Ve víceprocesorovém systému jsou běžící procesy přiděleny ke zpracování jednotlivým procesorům
    - Ty obvykle vykonávají různé části kódu a vyměňují si s ostatními procesory např. data a výsledky své činnosti
* **Multitasking**
  + V režimu multitasking je výpočetní čas procesoru počítače (CPU) rozdělen mezi současně spuštěné aplikace
  + Vzhledem k rychlosti procesoru vzniká dojem současného zpracovávání více úloh
  + Multitasking je vlastností operačního systému a pravým multitaskingem disponují až pokročilé systémy – MS Windows NT nebo systémy UNIX

## Harvardské schéma

* **Harvardská architektura má na rozdíl od von Neumanovy**
  + oddělený paměťový prostor pro data a pro instrukce
  + dovoluje používat pro paměť programu například paměti typu **ROM** (Read Only Memory)
  + umožňuje v podstatě zdvojnásobení velikosti paměti při stejně veliké adresové sběrnici
    - bit nutný pro realizaci této možnosti je obsažen v instrukčním souboru, který obsahuje instrukce pro komunikaci s datovou pamětí
    - u Von Neumanovy koncepce je paměť pro data a pro program společná - tímto způsobem se můžeme na instrukce dívat jako na data a během programu je můžeme měnit a ovlivňovat chování programu.
* **Harvardská architektura**
  + navržena **Howardem Aikenem** v třicátých letech minulého století na Harvardské univerzitě při vývoji reléového počítače HARVARD MARK 1 (1944)
  + dále byla využita na Pensylvánské univerzitě pro elektronkový počítač **ENIAC** (Electronic Numerical Integrator and Calculator)
  + již tehdy byla moderní koncepcí, ale technické prostředky v té době neumožnily její realizaci
  + byla zavržena – převzata koncepce von Neumannova
  + později asi po čtyřiceti letech dosáhla technologie výroby integrovaných obvodů takového stupně, že mohla být tato koncepce realizována
  + **Dnes běžně používána pro**
    - návrh signálových procesorů zvláště pro použití v telekomunikacích
    - při zpracování hudby a řeči
    - v medicíně
    - v průmyslu pro řízení a měření
* **Základní principy harvardské architektury jsou**
  + paměť je rozdělena na paměť programu a dvě paměti dat tak, aby současně mohly dva operandy vstupovat do ALU
  + součin dvou operandů v jednom instrukčním cyklu provede hardwarová násobička a výsledek součinu je přičten k akumulátoru (operace typu MAC - Multiply And Accumulate)
  + pro zvýšení výpočetní výkonnosti se používá **pipelining** (zřetězené zpracování instrukcí)
  + je zvýšen počet samostatných datových a adresových sběrnic
  + přímý přístup do paměti je prováděn vícenásobným kanálem **DMA** (Direct Memory Access)
  + řízení jádra procesoru je odděleno od řízení vstupních a výstupních jednotek
  + architektura typu 1X a 2X znamená, že jedna instrukce je provedena v jednom nebo ve dvou hodinových cyklech (taktech)

## Flynnova taxonomie

* základní myšlenkou zvyšování výpočetního výkonu vůči harvardské architektuře je zvětšování podílu paralelního zpracování
* pro klasifikaci počítačů se používá nejčastěji *Flynnova taxonomie*
  + dělí počítače podle toku instrukcí a proudů dat na čtyři hlavní typy
  + zavedl **M. J. Flynn** v roce 1966
* **systémy dělil** 
  + **podle počtu programů řešených současně**
    - ***SI***
      * *Single Instruction Stream*
      * v čase řešení problému běží jeden program
    - ***MI***
      * *Multiple Instruction Stream*
      * během řešení běží více programů paralelně
  + **podle počtu datových souborů zpracovávaných současně**
    - ***SD***
      * *Single Data Stream*
      * jeden zpracovávaný tok dat
    - ***MD***
      * *Multiple Data Stream*
      * více zpracovávaných toků dat
* Kombinací těchto možností vzniknou čtyři kategorie paralelních systémů:
  + ***SISD***
    - je chápán jako klasický von Neumannův počítač s jedním programem a jedním sériově přiváděným tokem dat
    - 1 procesor provádí 1 instrukční proud nad daty uloženými v 1 paměti
  + ***MISD***
    - v současnosti hypotetickou kombinací několika programů zpracovávajících jeden tok dat
    - takový princip se zatím nepoužívá a je někdy nesprávně zaměňován se systémem se zřetězeným zpracováním instrukcí (**pipelining**)
    - záměna je nedůsledná v tom, že pomocí zřetězeného zpracování instrukcí je aktivní vždy pouze jeden program
  + ***SIMD***
    - lze chápat jako větší počet funkčních jednotek pracujících na řešení téhož programu
    - všechny jednotky provádí současně tutéž instrukci, ale každá s jinými daty
    - 1 instrukce je prováděna nad množinou dat množinou procesorů
    - řídicí jednotka vysílá stejný kód ke všem procesorům a není tudíž třeba se starat o jejich synchronizaci
  + ***MIMD***
    - je obecný typ paralelního systému
    - je flexibilnější, ale mohou u něj nastat problémy se sladěním úrovní procesu i dat, nebo s alokací procesorů
    - množina procesorů současně provádí různé instrukce nad různými daty
    - obsahuje jednotky již tak samostatné, že každá z nich plní samostatný program a přitom zpracovávají jiná data
    - multiprocesory, multipočítače, distribuované systémy
* **z těchto čtyř typů jsou pro paralelní výpočty rozšířeny systémy SIMD a MIMD**
  + **u SIMD a MIMD systémů máme na vybranou, zda** 
    - dovolit každému procesoru přístup k celé paměti nebo
      * **sdílená paměť**
    - přidělit každému jen její část
      * **rozdělená paměť**
      * nenastávají kolize, ale je potřeba zřídit síť pro komunikaci mezi procesory, které jinak mohou komunikovat přes sdílenou paměť
      * pro zbudování sítě je instalován zvláštní hardware a software, což komunikaci zpomaluje

## Základní deska

* její vlastnosti určují do značné míry vlastnosti celého počítače, spojuje všechny další zařízení
* obvody umístěné na desce plošného spoje mainboardu slouží především pro podporu mikroprocesoru a sběrnic,
* konektory propojují jednotky umístěné mimo mainboard (pevné disky, operační paměti…)
* základní desky nemají stejné rozměry a nemusí jít umístit do každé skříně
  + většinou se ale dodržují dvě základní velikosti (ATX a microATX)
* úkolem výrobce je vyrobit desku co nejuniverzálněji, aby spolupracovala s co největším počtem dalších komponent
* **čipová sada**
  + stará se o komunikaci mezi jednotlivými částmi motherboardu
  + zaručuje spolupráci s dalšími komponenty počítače
  + pro každou ze třech hlavních skupin základních desek je k dispozici několik čipových sad
  + čipset limituje parametry periferií a komponent počítače, které se připojují k základní desce
  + ovlivňuje také výkonnostní profil sestaveného počítače
* k „dorozumívání“ desky (o vybavení na hardwarové úrovni) s operačním systémem slouží speciální program **BIOS**
  + umístěn v paměti typu ROM v DIP pouzdře
  + výstupy BIOSu jsou vzhledem k OS přesně definovány, vstupy od hardwaru jsou však různé
  + nejznámější výrobci BIOSů jsou
    - AMI (American Megatrens Incorporated),
    - Award
    - Phoenix
  + ke konfiguraci BIOSu slouží program SETUP
* **Propojky (jumpery)**
  + některé parametry základní desky je v zájmu univerzálnosti nutné nastavit nezávisle na BIOSu
  + toto nastavení se provádí pomocí **jumperů**
  + jumper je skupina kontaktů (pinů), které můžeme propojit, a nastavit tak určitou vlastnost mainboardu
  + propojení se provádí podle dokumentace k základní desce
* **Přepínače (switch)**
  + vyskytovaly se hlavně u starších desek
  + bývaly sdružovány do větších celků
  + sloužily také k definování vlastností desky pomocí polohy ON/OFF
* **Prvky základní desky**
  + patice ZIF (zero input force) pro mikroprocesor
  + patice s obvody DIP, v kterých je umístěna cache L2 (vedle mikroprocesoru)
  + baterie pro napájení CMOS
  + rozšiřující sloty PCI, dříve AGP a ISA
  + konektor integrovaného VGA adapter
  + dříve konektory integrovaného rozhraní I/O a integrovaného rozhraní IDE pro HDD a disketové mechaniky, dnes spíše SATA pro HDD a CD/DVD/BD atp. mechaniky
  + banky pro obvody operační paměti
  + dříve konektory myši a klávesnice PS/2, dnes spíše pouze větší množství USB
  + napájení základní desky (vzadu u konektoru klávesnice)
  + další neoznačené obvody pro podporu mikroprocesoru (nazývané chipset)

## Procesor

* slouží k zpracování instrukcí od programů, kterými je řízen
* některé operace zpracovává sám, k provedení některých dalších instrukcí používá různé komponenty počítače (např. operační paměť, disky, sběrnice, displej nebo tiskárny)
* základní vlastností mikroprocesoru je vytváření dat podle pokynů určitého programu
* **rychlost**
  + počet operací provedených za jednu sekundu
  + jednotka: **Hertz [Hz]**
  + podle hodinových kmitů generovaných krystalem na základní desce
* jádrem každého mikroprocesoru je **logický obvod**,
  + dokáže zpracovat sadu jednoduchých mikroinstrukcí
  + každý mikroprocesor je navíc vybaven **instrukční sadou**
    - programátorům poskytuje přívětivější prostředky pro napsání složitějšího aplikačního programu
    - převod instrukční sady na mikroinstrukce, které je mikroprocesor schopen řešit, obstarává **program napsaný v mikroinstrukcích**
      * ten je další podstatnou částí mikroprocesoru.
  + **dnes se uplatňují dvě koncepce mikroprocesorů**
    - liší se ve velikosti instrukční sady
    - nelze jednoznačně říci, která instrukční sada je lepší, vždy záleží na konkrétním typu řešeného problému
    - ***RISC***
      * *Reduced Instruction Set Computer*
      * obsahuje jen základní instrukce, které jsou jednodušší a rychleji proveditelné
      * uplatnily se ale jen v mikroprocesoru PowerPC (u IBM, Applu, Motoroly) a u velkých nebo jednoúčelových počítačů
    - ***CISC***
      * *Complete Instruction Set Computer*
      * mikroprocesor je vybaven co nejúplnější instrukční sadou
      * obsahuje i instrukce, které se používají velice málo
      * s ohledem na velký počet instrukcí pak bývá procesor příliš složitý a tím i dražší
      * u osobních počítačů převládají (u Intelu, AMD, Cyrixu)
      * stále více začínají uplatňovat prvky procesorů RISC
* každý mikroprocesor obsahuje **registry**
  + ty fungují jako vnitřní paměti mikroprocesorů, do kterých se ukládají momentálně zpracovávaná data
  + počet registrů a jejich přesné použití se u jednotlivých mikroprocesorů liší
* ke specifikaci adres v paměti, na kterých se nacházejí zpracovávaná data, se používají různé způsoby **adresovacích mechanismů**
* **instrukční sada mikroprocesoru** musí obsahovat instrukce pro
  + přesuny dat mezi pamětí a registry
  + aritmetické a logické funkce
  + instrukce pro řízení programu
  + několik systémových instrukcí
* mikroprocesor vyhodnocuje během své činnosti celou řadu přerušení, tzn. signálů
  + vysílají hardwarová zařízení nebo programy
  + snad všechny moderní mikroprocesory mají **vektorový systém přerušení**
  + vektory uložené v operační paměti, identifikované číslem přerušení, ukazují na adresu v paměti, kde je uložen obslužný program přerušení
  + aby se po zpracování přerušení mohl procesor vrátit ke své původní činnosti
    - ukládá svůj momentální stav do speciálního registru – do zásobníku
  + mikroprocesor také může přerušení dočasně zakázat.
* paměť spravuje mikroprocesor pomocí **jednotek správy paměti**,
  + v paměti překládají adresy tak, aby to bylo výhodné pro operační systém
  + zabezpečují také ochranu paměti
  + zabraňují programům v narušení běhu ostatních programů nebo operačního systému
* mikroprocesory dále obsahují paměť cache
  + mezisklad dat mezi různě rychlými komponentami počítače
  + účelem této paměti je vyrovnávání rychlostí v přenosu dat.
  + paměť **L1 cache** slouží k přechovávání dat ze sběrnice
  + na základní desce je instalovaná paměť **cache L2**,
    - optimalizuje přesuny dat mezi mikroprocesorem a operační pamětí
* schopnost mikroprocesoru zpracovávat posloupnost instrukcí se nazývá **architektura mikroprocesoru**
  + instrukce se zpracovávají buď **sekvenčně**, tj. jedna za druhou nebo
  + **superskalárně**, tzn. nezávisle na sobě.
* dalším zrychlujícím prvkem mikroprocesoru je **pipelining**
  + instrukce se zpracovávají najednou ve více fázích
* s ostatními komponentami počítače komunikuje mikroprocesor pomocí **sběrnic**
  + proudí jimi data, adresy nebo řídící signály (adresové, datové, řídící sběrnice)
  + čím více vodičů sběrnice má (definuje se tzv. „šířka sběrnice“), tím více dat je schopna přenášet
  + pokud má sběrnice 32 vodičů (sběrnice je 32bitová), může přenášet 232 různých informací
  + sběrnice se dělí na
    - adresové (obhospodařující paměť)
    - datové (přenáší data mezi mikroprocesorem a okolím)
* elektronické obvody tvořící mikroprocesor pracují v jednotlivých taktech, které udávají jejich pracovní tempo
  + každá základní deska je vybavena **generátorem taktů**, který generuje taktovací impulsy pro mikroprocesor
  + základní deska nabízí přepínání více taktovacích frekvencí
  + čím je vnitřní frekvence mikroprocesoru vyšší, tím rychleji mikroprocesor pracuje, ale také se více zahřívá
  + vnitřní frekvence je vždy násobkem frekvence vnější, která určuje rytmus práce periferních zařízení (=> pomalejší práce).
  + vnější frekvence je generována stejným časovým obvodem jako vnitřní, lze ji nastavovat umístěním jumperů (propojek).
* na základní desce lze upravit také **napájení mikroprocesoru**
  + napětí, kterým jsou procesory napájeny, se neustále snižuje, aby se snížilo jejich zahřívání vlivem spotřeby většího množství elektrické energie
  + velikosti napětí se pohybují kolem 3,3 V
  + mikroprocesory pro notebooky mají napětí nižší než 3 V
  + napájecí zdroj dodává na desku napětí vyšší (asi 5 V)
* je nezbytné procesory chladit pomocí **aktivních a pasivních chladičů**
  + pasivní chladiče se skládají z chladících žeber, kterými proudí vzduch
  + aktivní chladiče připojují ke konektoru napájecího zdroje

## Paměti

* **paměť je rozdělena na buňky** stejné velikosti, z nichž
  + každá je jednoznačně identifikována svým číslem (adresa paměti)
  + **velikost buňky** se označuje jako **nejmenší adresovatelná jednotka**
* **paměti je možné rozdělit na**
  + **vnitřní** neboli **operační**
    - slouží k uchovávání momentálně zpracovávaných dat a programů
    - realizovaná většinou polovodičovými součástkami (integrovanými obvody)‏
  + **vnější** neboli **periferní**
    - slouží k dlouhodobějšímu uchovávání dat a programů
    - realizovaná většinou na principu magnetického (popř. optického) záznamu dat, dnes také FLASH (SSD)
    - ve srovnání s vnitřní pamětí bývá obvykle pomalejší, ale levnější
    - pevný disk, CD-ROM, disketa, …
* **parametry**
  + **přístupová (vybavovací) doba**
    - doba, která uplyne od požadavku na čtení informací z paměti do okamžiku, v němž jsou data z paměti k dispozici
  + **přenosová rychlost** 
    - množství informací, které lze z paměti přečíst (nebo zapsat) za jednotku času
    - úzce souvisí se šířkou datové sběrnice
  + **kapacita**
    - množství informací, které lze do paměti uložit současně
  + **cena za bit**
    - určuje celkovou cenu paměťového systému
    - obecně platí, že rychlejší paměti mají vyšší cenu za bit uložených informací
  + **závislost obsahu na napájecím napětí**
    - zda se informace uložené v paměti po vypnutí napájení ztratí
* **paměť typu RWM (RAM)**
  + *Read/Write Memory*
  + pro čtení i zápis
  + po vypnutí napájecího napětí se obsah paměti vymaže
  + velice často se nesprávně označují jako, tj. paměť s náhodným přístupem
  + **rozlišujeme dva základní typy pamětí RAM**
    - **statické RAM** (SRAM)‏
    - **dynamické RAM** (DRAM)‏
* **statické RAM (SRAM)**
  + informace zapsaná do paměťové buňky zůstane zachována do té doby, než odpojíme napájecí napětí
  + oproti dynamickým RAM mají kratší přístupovou dobu, jsou však dražší a mají vyšší energetickou spotřebu
  + používají se především jako paměti typu **cache** (vyrovnávací paměť) a jako tzv. „paměť CMOS“
  + výrobní technologie **CMOS** má v klidovém stavu velmi nízkou spotřebu elektrické energie
  + proto se statické RAM CMOS používají pro **uchovávání konfigurace počítače a hodin reálného času**
  + paměť je při vypnutém počítači napájena malým akumulátorem nebo baterií
* **dynamické RAM (DRAM)**
  + informace zapsaná do paměťové buňky zůstává uchována jen po určitou dobu (řádově milisekundy)
  + potom musí dojít k jejímu obnovení (tzv. Refresh)‏
  + oproti SRAM jsou pomalejší a levnější
  + mají menší spotřebu a vyšší kapacitu
  + přístupová doba obvykle desítky nanosekund
  + používají se obvykle jako operační paměť počítače
  + **existuje několik typů dynamických RAM**
    - liší se svou rychlostí a činností
    - ***FPM*** – Fast Page Mode,
    - ***EDO*** – Extended Data Output,
    - ***SDRAM*** – Synchronous DRAM,
    - ***ECC*** – Error Checking and Correcting
* **paměti typu ROM**
  + paměť pouze pro čtení
  + obsah paměti je u klasické ROM určen již při výrobě
  + po vypnutí napájecího napětí zůstává obsah paměti zachován
  + ve srovnání s RAM je pomalejší, mívá menší kapacitu
  + v osobním počítači paměť typu ROM (dnes většinou Flash-EEPROM) obsahuje tzv. **BIOS**
  + nemožnost programování je velkou nevýhodou
  + **postupně se vyvinulo několik podtypů:**
    - **ROM**
      * klasická, obsah určen již při výrobě
    - **PROM** 
      * *Programmable ROM*
      * programovatelná, uživatel si ji může sám naprogramovat (jen jednou)‏
    - **EPROM**
      * *Erasable PROM*
      * lze opakovaně programovat, před každým programováním se však musí obsah paměti vymazat působením ultrafialového záření (asi ½ hodiny)‏
    - **EEPROM**
      * *Electrically EPROM*
      * maže se elektrickými impulsy
      * počet programování a mazání však bývá omezen
    - **Flash-EEPROM**
      * rychlejší než předešlé typy
      * dá se programovat přímo na desce

## Sběrnice

* sběrnici si můžeme obecně představit jako skupinu elektrických vodičů spojujících jednotlivé součásti počítače
* přes sběrnice probíhá komunikace mezi mikroprocesorem a periferiemi,
* **komunikovat s okolím může mikroprocesor třemi způsoby**
  + pomocí sběrnic
  + pomocí hardwarových přerušení (IRQ)
  + kanály přímého přístupu do paměti (DMA)
* sběrnic v počítači existuje hned několik (sběrnice procesoru, přídavných karet ap.)
* většina sběrnic je rozdělena na tři podsystémy:
  + **sběrnice datová**
    - přenáší pouze numerické údaje mezi zdrojovým a cílovým zařízením.
  + **sběrnice adresová**
    - přenáší adresy zařízení, které se účastní komunikace.
  + **sběrnice řídící**
    - přenáší signály pro řízení celé komunikace
* základním požadavkem na sběrnici je rychlost přenosu dat
* starší počítače byly konstruovány okolo jedné jediné sběrnice
  + to sice snížilo jejich cenu, ale také výkon
* na sběrnici mezi sebou mohly komunikovat jen právě dva prvky (vysílající a přijímající)
* ostatním částem není přístup na sběrnici (v dané chvíli) umožněn
* sběrnice musí vyhovovat typu mikroprocesoru,
  + udává šířku datové i adresové části sběrnice
* **režimy využívané sběrnicemi**
  + **multimastering**
    - provoz sběrnice řídí některá z přídavných karet, která provádí některé úkoly mikroprocesoru
    - procesor se tak nemusí zatěžovat komunikací přes sběrnici a prací některých periférií.
  + **burst režim**
    - data jsou přenášena po skupinách bez adresy u každého datového bloku
    - urychlení
  + **lokální sběrnice**
    - současné počítače jsou postaveny na hierarchické sběrnicové struktuře
    - základní je tzv. lokální sběrnice
    - po ní musí komunikace probíhat nejrychleji, protože na ni vstupuje procesor, který je nejrychlejší součástí počítače
    - dokonce ani konstrukce dnešních sběrnic není na takové úrovni, aby vyhovovala výkonu procesoru
    - proto se do taktu procesoru vkládají tzv. čekací cykly
      * tyto cykly zajišťují bezchybnou komunikaci procesoru se sběrnicí
      * zároveň snižují jeho výkon.
  + **rozšiřovací sběrnice** 
    - **systémová**
    - odděluje mikroprocesor od “okolního světa”
    - je cestou pro připojení dalších komponent (většinou od různých výrobců)
    - sběrnice vyúsťuje konektory, jednotlivými sloty, do kterých se přídavné karty zasazují.
    - u systémové sběrnice jsou kladeny nejvyšší nároky na rychlost přenosu a na důslednou standardizaci (kompatibilitu)
      * aby do slotů (patlí, zástrček) sběrnice mohly být instalovány přídavné karty různých výrobců
      * rozšiřovacích sběrnic je u počítačů IBM PC kompatibilních hned několik druhů,
      * liší především svou propustností (množstvím dat, která projdou přes sběrnici za jednotku času)
* **XT BUS** 
  + stará osmibitová sběrnice
* **osmibitová ISA**
  + *lndustry Standard Architecture*
  + nejstarší varianta sběrnice osobních počítačů PC XT (před 80286)
  + stará a relativně pomalá sběrnice
  + vyskytuje ve dvou variantách (starší osmibitová a novější šestnáctibitová)
* **šestnáctibitová ISA**
  + zdokonalená verze pro počítače PC AT a 80286
  + má 16bitové rozšíření
  + počet karet připojených přes sběrnici není už omezen
  + slot sběrnice bývá hnědý nebo černý
  + v BIOSu lze zvýšit její kmitočet, nastavení pomocí jumperů
* **MCA**
  + *MicroChannel Architecture*
  + sběrnice vyvinutá firmou IBM pro počítače řady PS/2, 80386
  + není kompatibilní s ISA
* **EISA**
  + *Extended ISA*
  + jak název napovídá, jedná se o zdokonalení dřívějšího standardu ISA
  + kompatibilní s *ISA*
  + stále však nízká frekvence
  + slot sběrnice je dvoupatrový (vytvořený ze starých ISA slotů)
  + konfiguruje se programově
  + nedosáhla velkého rozšíření
* **VL-BUS** / **VESA VL-BUS**
  + *Video Local Bus*
  + standard vytvořený v roce 1991 několika výrobci základních desek
  + předchůdce PCI, používala se hlavně u starších mikroprocesorů 80486
  + je koncipována jako rozšíření ISA
  + VL-BUS není klasickou sběrnicí, je jen nástavbou ISA
  + nevýhodné je, že s rostoucí frekvencí rychle stoupá zatížení sběrnice
  + mohlo se použít jen pár slotů, a proto se sběrnice také nerozšířila
* **PCI**
  + *Peripheral Component Interconnect*
  + sběrnice definovaná v roce 1993 firmou Intel v souvislosti s nástupem Pentií
  + vyznačovala se vysokou přenosovou rychlostí (nad 100 MB) a spolehlivostí, nezávislostí na frekvenci procesoru
  + sběrnice přinesla také normu **PnP (plug-and-play)**
  + **k**vůli zatížení nemůže být na desce více než 3-4 PCI slotů
* **AGP**
  + 32bitová
  + oproti PCI průchodnější pro paměti
  + podpora grafických akcelerátorů - často se na ni napojují videokarty
* **procesorová sběrnice FSB**
  + *Front Side Bus*
  + jedná se o nejrychlejší sběrnici celého systému
  + tvoří také jádro základní desky a čipové sady
  + sběrnice je využívána především samotným procesorem k práci s cache a systémovou pamětí
  + u soudobých PC tato sběrnice pracuje na rychlosti 66, 100, 133 nebo 200MHz
  + má šířku 64 bitů
* Dále viz otázka 4

## řadič

* Elektronická řídicí jednotka, realizovaná sekvenčním obvodem, která řídí činnost všech částí počítače. Řízení pomocí řídicích signálů, které jsou zasílány jednotlivým modulům (částem počítače). Reakce na řídicí signály – stavy jednotlivých modulů – jsou naopak zasílány zpět řadiči pomocí stavových hlášení.

## přídavné karty

* Slouží pro doplnění a rozšíření konfigurace počítače (při nižších nárocích, lze pořídit následující komponenty přímo na základní desce)
* **Grafická karta** (adaptér)
  + Zabezpečuje zobrazení informací na zobrazovací jednotce (monitor)
  + Obsahuje grafický procesor (GPU – hlavní výrobci AMD, dříve ATI, a nVidia) a volitelně grafickou paměť (DDR2, GDDR3, GDDR4)
  + Zpracovává informace z CPU a posílá na výstup (D-SUB, DVI, HDMI, DisplayPort)
  + Dnes připojována přes PCI Express, dříve přes AGP
  + Lze paralelně propojit více karet ke spolupráci – technologie SLI (nVidia) a CrossFire (AMD)
* **Zvuková karta**
  + Obstarává zvukový vstup a výstup, skládá se z A/D převodníku, D/A převodníku, zesilovače, filtru atd.
  + Dnes připojována přes PCI, dříve přes ISA.
  + Standardizované barevné značení: např. zelená – LINE-OUT, růžová – MIC, modrá – LINE-IN
* **Síťová karta**
  + Pro vzájemní propojení počítačů v síti
  + Obvykle konektory pro připojení RJ45, případně optika nebo pouze Wi-Fi (může být připojitelná anténa přes koaxiální kabel)
  + Připojována přes PCI, u starších notebooků též přes PCMCI
* Aj.

## Ovladače

* **Device driver** – software, který umožňuje operačnímu systému pracovat s hardwarem
* Některé ovladače součástí OS, jiné dodávány s HW

|  |
| --- |
| Aplikační vrstva |
| Operační systém |
| Ovladač zařízení |
| Hardware |

# Paměťový systém počítače a ukládání dat

Typy, principy fungování, frekvence, normy, logická a fyzická struktura disku, RAM, ROM, Cache, HDD, CD, DVD, FLASH

## Typy

* **podle fyzikálního principu** použitého pro uložení dat můžeme média rozdělit na
  + **magnetická** (disky, diskety)
  + **optická** (CD a DVD)
  + **elektrická** (Flash disky, paměťové karty)
* **pružné disky (FD, Flopy Disk, diskety)**
  + mají velikost **3.5"**, dříve **5.25"**
  + disketu i pevný disk je nutno před prvním použitím naformátovat příkazem
  + počítač vytvoří na magnetickém povrchu "značky", podle kterých později zapisuje data
  + povrch je rozdělen na stopy (soustředné kružnice), resp. válce, a každá stopa je rozdělena na sektory (výseče)
  + do jedné výseče se zapisuje 512 bajtů
  + v současnosti používané diskety jsou oboustranné
  + standardní způsob formátování diskety 3.5" HD je tento:
    - 2 strany x 80 stop x 18 sektorů x 512 B = 1474560 B = 1440 KB = 1.41 MB
    - tato disketa se nepřesně označuje 1.44 MB
  + diskety s kvalitnějšími povrchy lze formátovat na ještě větší kapacity
* **pevný disk** **(HDD, Hard Disk)**
  + pevné disky se skládají z více částí, takže mají několik povrchů a kapacity v řádu stovek GB
  + moderní BIOSy mají autodetekci disků
  + pevný disk můžeme rozdělit na několik částí (partitions)
  + jednotlivé části se potom formátují nějakým souborovým systémem
    - novější NTFS formátování umožňuje nastavovat přístupová práva k jednotlivým souborům a složkám na disku
  + dlouhodobým používáním disku či diskety vzniká **fragmentace** souborů
    - soubory jsou rozděleny do několika nesouvislých částí, což zpomaluje jejich čtení
    - k odstranění fragmentace se používá program **DEFRAG**
    - pro vyhledávání chyb na disku slouží program **SCANDISK**
* **k ukládání a distribuci dat se používají i optické disky**
  + **CD-ROM** 
    - optické disky s kapacitou od 650 MB, které nelze přepisovat
    - klasické CD-ROM se vyrábějí lisováním stejně jako běžné hudební CD (proto na ně nelze zapisovat)
    - rychlost CD mechanik se udává v násobcích rychlosti 150KB/s
    - na rozdíl od pevného disku, který má soustředné kruhové stopy rozdělené do sektorů, má CD-ROM záznamovou stopu ve tvaru spirály začínající u středu disku, která je také rozdělena na sektory
    - spirálovitá stopa má stoupání 1,6 m, šířku 0,6 m a je tvořena různě dlouhými prohlubněmi (tzv. pity)
    - záznam je snímán laserovým paprskem
      * laserový paprsek přes soustavu čoček dopadá na plochu se záznamem, která je opatřena reflexní vrstvou
      * prohlubně zmenšují intenzitu odraženého světla
      * odražený laserový paprsek dopadá na fotocitlivý prvek, který intenzitu dopadajícího světla převádí na elektrický signál
  + **CD-R - (Recordable)**
    - CD disky, na které lze zapisovat
    - jednou zapsaná data již nemohou být přepsána
  + **CD-RW - (ReWriteable)**
    - CD disky, na které lze zapisovat a lze i přepisovat starší data novými
  + **DVD - (Digital Versatile Disc)**
    - optický disk s vysokou hustotou záznamu
    - vychází z technologie CD
    - odlišnosti média DVD od klasického CD:
      * hustější záznam – jiná rozteč stop
      * dvě záznamové vrstvy nad sebou – horní vrstva polopropustná, spodní vrstva odrazová; horní i spodní vrstva se čtou pouhým přeostřením laseru
      * oboustranný disk
    - vývoj DVD původně v oblasti videotechniky
    - svět rozdělen celkem na 6 regionů, ČR region 2
    - každé DVD s filmem určeno pro určitý region, lze přehrávat pouze na přehrávači nastaveném pro tento region
    - podle počtu povrchů a vrstev rozdělujeme 4 typy
      * Označení -Počet stran-Počet vrstev-Celková kapacita
      * **DVD-5**-1-1-**4,7 GB**
      * **DVD-9**-1-2-**8,5 GB**
      * **DVD-10**-2-1-**9,4 GB**
      * **DVD-18**-2-2-**17 GB**
  + **Blu-ray**
    - stejné rozměry jako CD a DVD
    - kapacita až 25 GB na 1 vrstvu (max 4 vrstvy)
* velmi populární jsou i **USB Flash disky** 
  + velmi malá zařízení připojovaná přes USB s kapacitou v GB
  + princip práce se nejvíce podobá pamětem RAM, ale data si pamatují i po odpojení napájení
  + nejčastěji NAND typ

## Pevný disk

* HDD i disketová mechanika obsahují tyto základní části:
  + médium s daty
  + magnetické hlavy pro čtení/zápis
  + mechaniku pohybující s hlavami
  + motorek točící diskem
  + řadič (=elektronický obvod, který řídí práci disku)
  + rozhraní zajišťující připojení disku k základní desce
* **princip záznamu a čtení**
  + informace je uchována pomocí změny elektromagnetického potenciálu média
  + s tím rozdílem, že nedochází k přímému dotyku hlavy s médiem
  + magnetická hlava se pohybuje v minimální vzdálenosti (mikrometry) nad povrchem disku
    - plave na vzduchovém polštáři vytvořeném aerodynamickým vztlakem
* jejich povrch je tvořen magnetickou vrstvou, která je přikryta vrstvou skleněnou
* přenos dat z elektromagnetické hlavy je digitální o rychlosti několika desítek MB/s
* rychlost rotace pevného disku se zvýšila z 3600 na 7200 a pak i přes 10000 otáček za minutu
* **logická struktura**
  + logická struktura je vytvořena vysokoúrovňovým formátováním, které umožňuje každý operační systém
  + je popisována soustavou tabulek
  + FAT32, NTFS…
* **fyzická struktura**
  + skládá se z tuhých **kovových disků** (tzv. propojovací desky, plotny)
  + jsou potaženy povlakem oxidu železa a uloženy na sebe ve vzduchotěsně uzavřeném prostoru
  + pro obě strany každé desky má HD **elektromagnetickou čtecí a zápisovou hlavu**, pod kterou se propojovací deska otáčí
  + hlav je dvakrát více než ploten (pro každou stranu plotny, pokud jsou oboustranné)
  + čtení dat bylo založeno na magnetické indukci, dnes už však funguje dokonalejší systém
  + data jsou čtena jako sled změn odporů vyvolaných různou orientací magnetického pole (magnetických dipólů)
  + všechny hlavy disku jsou napevno připojeny k **vystavovacímu raménku**, takže nemohou být nastaveny nezávisle na sobě a pohybují se současně
  + při vypnutí disku zajistí **mechanika magnetických hlav** jejich přistání do vyhrazené parkovací oblasti
    - hlava se nikdy nesmí dotknout disku, a poničit tak data
  + při náhlém výpadku proudu se hlavy samovolně vrací do parkovací zóny
* velikost HD je stejně jako u disket *5,25"* nebo *3,5"*, popř. dnes i *2.5"*
* fyzicky je disk rozdělen do tzv. **sektorů**
  + tyto sektory jsou spojeny do **soustředných stop** – kružnic
  + na jednu plotnu připadá 305 a více stop. Počet sektorů na stopu je u HD 17, 26, 34 nebo 52 a více
  + podle kódování dat dělíme HDD na RLL, MFM a PRML disky
  + rozdělení na stopy a sektory vzniká při fyzickém formátování disku
* **rychlost disku**
  + rychlost HD posuzujeme podle **přístupové doby** (tj. doba trvání vyhledání dat)
  + přístupová doba se skládá z
    - **doby vystavení**
      * =doba vyhledávání, Track-to-Track Seek
      * je výrobcem definována jako 1/3 doby potřebné pro pohyb hlavy přes celý disk
    - **doby čekání** (celkem kolem 10 ms, disk je tedy asi o dva řády pomalejší než operační paměť)
      * udává dobu, než se pod hlavu dotočí příslušný sektor, a závisí tedy víceméně na náhodě
      * technicky je uvažována jako jedna polovina otáčky disku
  + rychlost se dále posuzuje podle **rychlosti přenosu dat** (doba přečtení dat v MB/s).
* [**diskové rozhraní**](http://infweb.webpark.cz/Images/hddWD.jpg) 
  + standard komunikace mezi diskem a řadičem pevných disků
  + nejznámější diskové rozhraní je **IDE** (Integrated Device Electronics)
    - respektive EIDE Enhanced IDE (= ATA 2. generace)
    - pracující se standardem **ATA** (Advanced Technology Attachment) a z něj vycházející **Ultra ATA/33, Ultra ATA/66, Ultra ATA/100, Ultra ATA/133**
    - zcela specifické rozhraní je **SCSI**
    - specifické je i rozhraní **ATAPI** (ATA Packet Interface)
      * umožňuje rozhraní IDE komunikovat i se zařízeními jako CDROM apod.
      * nejedná se o samostatné rozhraní, ale jak z názvu vyplývá, jedná se o rozšířené rozhraní **ATA**
    - **PIO** (Processor Input/Output)
      * komunikační protokoly určující rychlost komunikace
      * rozlišujeme módy PIO, kdy je **přenos řízený procesorem**, a **DMA** (Direct Memory Access), který přistupuje přímo do paměti.
    - rozhraní **Serial ATA**
      * rozhraní neposílá data paralelně plochým 80tižilovým kabelem ale přes tenký kabel
      * zlepší i cirkulaci vzduchu ve skříni
      * každý disk má svůj vlastní kabel, takže se nemusí dělit
      * nové rozhraní užívá **DMA** kanály
      * datový tok je 150MB/s v případě SerialATA 1x a až 600MB/s u Serial ATA 4x
* **Master, Slave**
  + IDE řadič umožňuje připojit 2 zařízení na jeden kabel
  + aby se zařízení nehádaly při komunikaci, nastaví se jedno z nich jako master
    - řídí komunikaci na kabelu
  + druhé zařízení se pak podřídí (slave)
  + k tomuto nastavení slouží jumpery umístěné na daném zařízení

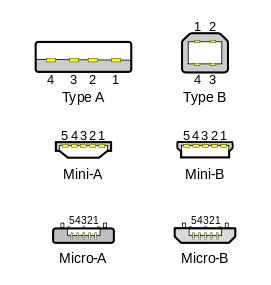
# ARCHITEKTURA PERIFERNÍCH ZAŘÍZENÍ

Rozdělení, principy, funkce, typy, rozhraní, příklady

* Zařízení, které není nezbytně nutné k provozu počítače, které však rozšiřuje jeho schopnosti. V širším slova smyslu se za periférii považuje cokoli kromě základní desky počítače s jeho procesorem (tj. paměť, disk, disketová mechanika, porty, klávesnice, monitor), v užším slova smyslu zařízení připojovaná k počítači externě, která nejsou nezbytně nutná k běžnému provozu (např. tiskárna, plotr, skener, modem, síť atd.).
* Rozdělení dle způsobu připojení skrze externí nebo interní rozhraní a na vstupní a výstupní.
* *Pozn.: periferie = spíše jen ty externí, interní = komponenta, nicméně komunikace s PC probíhá stejným způsobem*

## EXTERNÍ ROZHRANÍ

* **PS/2** (klávesnice a myš)
  + uvolnění nedostatkových portů COM a snížení nákladů na výrobu periférií (komunikace probíhá 0/+5V – není třeba převod na RS232, a vyvedené napájení)
  + Ve starších PC konektor DIN který má stejné signály jako konektor PS/2.
* **VGA** (monitor CRT či LCD)
  + standardně pro připojení zobrazovacího zařízení.
* Rozhraní zvukové karty
  + Zelená – zvukový výstup (reproduktory)
  + Modrá – Line-in vstup (například pro nahrávání z externích zařízení)
  + Oranžová – mikrofon
* Rozhraní síťové karty
  + Rozhraní sloužící pro propojení PC s jiným nebo pro připojení PC do lokální počítačové sítě
  + **BNC** (koaxiál, 50 ohmů, sériové zapojení)
  + **TP** (či **UTP**, dle kategorie a rychlosti sítí)
  + **FX** (optika)
* Paralelní port **LPT**
  + tiskárny, paralelní skenery, laplink – spojení dvou PC
  + na PC zakončen konektorem canon 25pin samice, rychlejší než COM díky paralelnímu přenosu
* **USB**
  + velmi populární, podpora P&P, možnost připojení více zařízení na jeden port a napájení přímo z PC
  + USB 1.1 (teoretická rychlost 12 Mb/s, pomalé pro přenos)
  + USB 2.0 (teoretická rychlost 480 Mb/s)
  + USB 3.0 (teoretická rychlost až 5 Gb/s)
  + typ A (výstupní – na PC, rozbočovače)
  + typ B (vstupní – modemy, tiskárny)



* **FireWire**
  + méně rozšířené oproti USB, není jednoznačný název (IEEE 1394, iLink atp), standardizováno skupinou IEEE
  + znatelné rozšíření hlavně v oblasti zpracování videa, přenosová rychlost 400 Mb/s
* Sériový port (**COM**)
  + jeden z nejrozšířenějších standardů RS232, využití i jako laplink
  + zakončen konektorem Canon 25pin nebo 9pin samec
  + připojení polohovacích zařízení, modemů, měřících přístrojů, zařízení se sériovým I/O.
* Bezdrátové rozhraní
  + **IrDA**
    - standard pro bezdrátový přenos digitálních dat pomocí infračerveného záření
    - potřeba mobilně spojit různá zařízení (mobily, notebooky, PDA, videokamery atp.)
    - specifikace (standard) pro fyzická koncová zařízení i komunikující protokol (několik)
  + **WiFi** (Wireless Fidelity)
    - jeden ze standardů (IEEE 802.11\*) bezdrátové komunikace
    - vzájemná bezproblémová komunikace různých výrobců, různá pásma, rychlosti a dosah

|  |  |  |  |
| --- | --- | --- | --- |
| Standard | Rok vydání | Pásmo [GHz] | Maximální rychlost [Mbit/s] |
| původní IEEE 802.11 | 1997 | 2,4 | 2 |
| IEEE 802.11a | 1999 | 5 | 54 |
| IEEE 802.11b | 1999 | 2,4 | 11 |
| IEEE 802.11g | 2003 | 2,4 | 54 |
| IEEE 802.11n | 2009 | 2,4 nebo 5 | 600 |
| IEEE 802.11y | 2008 | 3,7 | 54 |
| IEEE 802.11ac | 2013 | 5 | 1000 |
| IEEE 802.11ad | 2014 | 2,4, 5 a 60 | 7000 |

* + - Přenos ovlivňuje: vzdálenost, podmínky, výkon antény, zástavba, rušení.
    - Stále rostoucí tlak na zvyšování přenosové rychlosti > standardy 802.11g a 802.11a.
  + **Bluetooth**
    - radiové vlny o frekvenci 2,45 GHz procházejícími většinou materiálů.
    - problém Faradayovi klece (železo-betonové budovy)
    - pro point-point max. vzdálenost 10m, pro point-multipoint vzdálenost 100m
    - spojení plně duplexní (příjem i vysílání současně)
    - stejné pásmo jako IEEE 802.11 (rušení, interference), cca 10 x pomalejší a menší dosah.

## INTERNÍ ROZHRANÍ (SBĚRNICE)

* **Sběrnice**
  + paralelní soustava vodičů, sběrnici (**BUS**) můžeme definovat jako komunikační standard k propojování různých zařízení s cílem umožnit mezi nimi přenos dat, případně i elektrické energie
  + za řízení přenosu dat po sběrnici zodpovídá její **řadič**
  + **Parametry**
    - šířka přenosu – počet bitů, které lze zároveň po sběrnici přenést, (bit)
    - frekvence – maximální frekvence, se kterou může sběrnice pracovat (Hz)
    - rychlost (propustnost) – počet bytů přenesených za jednotku času (B/s)
* **Typy rozdělení**
  + Způsob komunikace – Paralelní, Sériová
  + Skupina vodičů („funkce“) – Datová, Adresová, Řídící, Napájecí
  + Přenos signálu – Synchronní, Asynchronní
  + Výkon – Datová propustnost, Řídící kmitočet
* **Standardy sběrnic**
  + **ISA**
    - 8b/16b sběrnice používané u IBM PC XT
  + **PCI**
    - sběrnice vyvinutá firmou INTEL pro systémy postavené na bázi mikroprocesoru PENTIUM
    - Je důsledně oddělena od lokální sběrnice procesoru, pracuje proto na rozdílné frekvenci 33 MHz.
    - PCI-X pro rozšiřující karty a PCI-Express – sériově, pomocí paketů.
  + **AGP**
    - Pro přenos dat do zobrazovací soustavy
    - V počítači pouze jeden, teoretická propustnost až 533MB/s
    - Speciální sběrnice propojující grafický adaptér s operační pamětí – Advanced Graphics Port.
  + **SCSI**
    - rozhraní a sada příkazů pro výměnu dat mezi externími nebo interními počítačovými zařízeními a počítačovou sběrnicí (běžně pro HDD, ale také CD-ROM, DVD či skener).
    - využívá se většinou u výkonných stanic (serverů s RAID disky téměř vždy přes SCSI).
    - na SCSI sběrnici jsou napojena jednotlivá zařízení, která jsou si v připojení rovnocenná
    - jedním zařízením je vlastní SCSI řadič (4 standardy, rozdíly hlavně v rychlostech).
    - dnes hlavně Ultra320 SCSI - Rychlost 320 MB/s, Ultra640 SCSI - Rychlost 640 MB/s.
  + **IDE/ATA**
    - standardy elektronického rozhraní mezi počítačem a jeho jednotkami velkokapacitní paměti.
    - IDE je označení pro ATA 1, EIDE = ATA 2, dále až ATA/ATAPI 7 – rozdíly v rychlosti a maximální podpoře paměti (max. 133MB/s a 144PB)
    - Připojení dvou zařízení na jeden kabel
    - zapojení dle priority jako Single/Master/Slave/CSEL1
  + **SATA** (SerialATA)
    - na jeden kabel jedno zařízení, sériová komunikace
    - vychází z ATAPI 7
    - SATAI (1,5 Gb/s), SATAII (3 Gb/s), SATAIII (6 Gb/s).

## VSTUPNÍ PERIFERIE

* **Myš**
  + polohovací zařízení (většinou dvourozměrně), dříve kulička, dnes optické (a přesnější laserové) snímání na podložce – CCD čidlo
  + alternativou trackball, touchpad, tablet (malířský, připojený přes USB, bez displeje, ale i s)
  + parametrem zejména rychlost snímače (DPI)
  + rozhraní: PS/2, USB, bezdrátově (někdy připojena i ke klávesnici)
* **Klávesnice**
  + nejdůležitějším vstupním zařízením počítače. (staré XT a „nové“ AT)
  + standard 101 kláves + klávesy speciální pro OS (Win – Start, Kontext)
  + alfanumerické znaky, numerická klávesnice, funkční klávesy (F1-F12) + multifunkce
  + parametry: prodleva stisknutí klávesy, rychlost opakování znaku, rozložení znaků
  + rozhraní: DIN, COM, PS/2, USB, bezdrátové
  + při stisku klávesy klávesnice předává „kód klávesy“, ovladač/řadič přidělí odpovídající znak.
* **Scanner**
  + předloha je po řádcích osvětlována a odražené světlo je optickým systémem vedeno k přijímači.
  + typy skenerů: ruční (300-400dpi a 256 barev, „amatérské“), stolní (až 1200dpi a 32bit barev), bubnový (až 4000dpi a 48 bitů barev na pixel)
* **Herní zařízení**
  + GamePad
    - dříve připojeno přes GamePort (žlutý, rozměrově něco mezi COM a LPT)
    - nyní výhradně USB
  + Joystick
    - pákový ovladač – imitace kniplu neboli řídící páky letadla
  + Volant
  + 3D kamery a polohovací zařízení
    - Kinect aj. (kombinace dvou a více kamer, skládání a rozpoznávání obrazu)
    - Boom ovládání pohybem
* **Webová kamera**

## VÝSTUPNÍ PERIFERIE

* **Tiskárny** a **plotry**
  + plotr – projekční činnost pro tisk vektorové grafiky, pohyb hlavy po více osách, případně více hlav atp.
  + tiskárna – obyčejný tisk a bitmapová grafika
  + typy tisku: jehličkový, řádkový, inkoustový, laserový, termotransferový, termosublimační
  + parametry: DPI, barevnost, rozměr stránky, rychlost, komunikační rozhraní, podavače
* **Zobrazovací zařízení**
  + rozdělení dle použití technologie
  + **CRT** (Cathode Ray Tube)
    - tři katody emitující paprsky elektronů (obraz z RGB)
  + **LCD** (Liquid Crystal Display)
    - 2 desky pokryté elektrodami, mezi nimi tekuté krystaly, podsvíceno soustavou zářivek po obvodu panelu + vrstva rozvádějící světlo
    - V současnosti moderní nahrazení zářivek LED podsvícením
  + **Plazma**
    - dvě elektrody, mezi nimi plyn (směs argonu, xenonu, neonu)
    - dříve lepší vlastnosti (zobrazení) než LCD, dnes vyrovnané, nevýhodou je velký odběr energie (vysoké napájecí napětí, viz TV)
* **Zvuková zařízení**
  + buď přídavná karta, nebo na základní desce
  + záznam, přenos a reprodukce zvuků
  + použití: digitalizace analogové hudby, komponování hudby, přehrávání hudby
  + fyzické složení: digitální část (A/D,D/A převodník), syntezátor (zdroj zvuku), rozhraní MIDI
  + parametry: stereo/mono výstup, vzorkovací frekvence, počet bitů vzorkování, MIDI systém
  + využití: hry, součást prezentace, reprodukce hudby, síťové konference, multimédia

# Komunikační prostředky

Principy komunikace, modulace signálu, rozdělení a porovnání, média, mobilní technologie

## Principy, typy a porovnání

Co je potřeba ke komunikaci:

* Společný jazyk (protokol)
* Navázaní komunikace
* Řízení komunikace (kdo vysílá a kdo poslouchá)
* Adresa cíle a adresa zdroje

**Typy komunikace**

Paralelní vs sériová:

* V paralelní komunikaci se odesílá několik dat najednou, kdežto v sériové se posílají data postupně za sebou.
* Paralelní komunikace vyžaduje synchronizaci pomocí hodin, aby bylo jasné která data jsou platná
* Dříve se používala spíše paralelní komunikace dokud nenarazila na limity, kdy už nešlo zvyšovat frekvenci aniž by se zvýšil počet chybně doručených dat.
* Dnes se používá zejména sériová komunikace.
* Sériová komunikace umožňuje mít menší a delší kabel a není nutné řešit problémy se synchronizací
* Navíc sériová komunikace umožňuje použití vyšších frekvencí
* PCI vs PCI-e

Synchronní vs asynchronní:

* Synchronní komunikace je komunikace v reálném čase
* Při synchronní komunikace zařízení aktivně čeká na komunikaci
* Synchronní komunikace – telefonní hovor
* Asynchronní komunikace nevyžaduje okamžitou odezvu – email
* Asynchronní komunikace umožňuje aby se zařízení nezdržovalo čekáním na data.

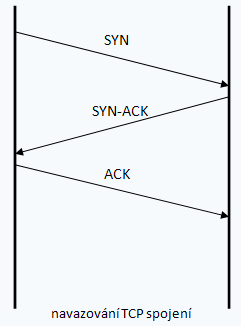
Znehodnocení signálu:

* Útlum
  + Vzniká přirozenou ztrátou síly signálu v závislosti na délce kabelu
* Šum
  + Je rušivý signál, který v průběhu komunikace mění a poškozuje přenášenou zprávu.
  + Dochází tím k různě významnému zkreslení informace

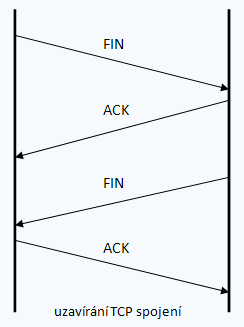
Počítačová síť a její prvky:

* + **Zařízeními** – počítače, smartphony, směrovače, družice, tiskárny
  + **Přenosovými médii** – metalické a optické kabely, prostor
  + **Protokoly** – stanovenými pravidly komunikace, formáty datových struktur, výměny řídících a kontrolních informací
  + **Zprávami** – přenášenými daty spolu s řídícími informacemi
  + **Topologie** – způsob zapojení jednotlivých zařízení
    - **Fyzická topologie** – zachycuje reálné zapojení pomocí drátů a kabelů
    - **Logická topologie** – zachycuje vnitřní (virtuální) propojení, nemusí nutně kopírovat fyzickou topologii

Komunikační protokoly musí zajistit minimálně:

* detekci fyzického spojovacího média, detekci uzlů, koncových zařízení
* handshake – nastavení parametrů spojení mezi komunikujícími zařízeními
* vyjednávání o parametrech spojení
* označení a detekce začátku a konce zprávy
* formátování zpráv
* detekce chyb
* způsob nápravy chyb
* ukončení spojení

TCP protokol - navázání spojení:

1. Strana zahajující spojení (klient) vyšle TCP segment s nastaveným příznakem SYN.
2. Strana přijímající spojení (server) odpoví TCP segmentem s nastavenými příznaky SYN+ACK.
3. Klient odpoví TCP segmentem s nastaveným příznakem ACK

TCP protokol – ukončení spojení:

***čtyřcestný handshake***, kdy každá strana samostatně uzavře spojení. Zde se používá sekvence FIN s odpovědí ACK.

Komunikace peer-to-peer vs client/server

* Klienti komunikující vzájemně mezi sebou – P2P
* Klienti komunikují prostřednictvím serveru – C/S

Broadcast – odeslání zprávy všem zařízením v síti

Duplexní spojení

* Half-duplex
  + Obě strany mohou přijímat i vysílat, avšak nikoli současně – v každý jednotlivý okamžik probíhá přenos pouze jedním směrem (podobné simplexu).
  + Při této komunikaci jsou však na rozdíl od simplexu využívány dvě frekvence.
  + Na jedné frekvenci se vysílá, na druhé přijímá.
  + Příkladem takové komunikace je vysílání radiostanic (vysílaček) přes opakovač; typické pro half-duplexní spojení je používání signalizace „přepínám“.
* Full-duplex
  + U plného duplexu může obousměrná komunikace probíhat současně.
  + Příkladem takové komunikace může být běžný telefonický hovor, kdy obě zúčastněné strany mohou hovořit zároveň.
  + Plný duplex v Ethernetu (varianta 100Base-TX) funguje tak, že jeden pár vodičů v ethernetovém kabelu je využívány pro odesílání rámců a druhý pár je využívány pro příjem. Zbylé dva páry vodičů nejsou využity.

## modulace signálu

* změna původní charakteristiky signálu na charakteristiku vhodnou pro přenos skrze přenosové prostředí
* využívá se pro přenos rozhlasových a televizních vln, v mobilní komunikaci, v satelitních přenosech
* **přenos signálu**
  + v základním pásmu (*baseband*)
  + v modulovaném pásmu (*broadband*)
* **základní pásmo**
  + bez modulace
  + náchylný k rušení
  + unipolární (0V = 0, >0V = 1)
  + bipolární (-5V = 0, >0V = 1)
  + kódování Manchester (změna z vyšší na nižší = 0 a naopak)
  + nelineární proces, kterým se mění charakter vhodného nosného signálu pomocí modulujícího signálu
  + používá se při přenosu nebo záznamu elektrických nebo optických signálů
* **modulační signál**
  + signál na vstupu modulátoru, který bude modulován do vhodné charakteristiky
* **nosný signál** 
  + signál, který „ovlivní“ signál na vstupu modulátoru
* **modulovaný signál**
  + signál na výstupu modulátoru, určený pro finální přenos
* **změna tvaru nosného signálu**
  + amplitudová
  + frekvenční
  + fázová modulace
* **digitální modulace s nosnými vlnami**
  + kvůli optimalizaci přenosu
  + modulace na vysokofrekvenční složky
  + tento způsob umožňuje přenášet více informačních signálů po jednom komunikačním kanálu
  + diskrétní signály nelze přenášet rádiovým přenosem
    - digitální modulace => lze vzduchem

## Komunikace

* předávání informací
* **formy komunikace** 
  + „lidská“
    - verbální
    - neverbální
  + datová
* **informace**
  + stěžejní obsah komunikace
* pro přenos informací se využívají přenosové cesty
* **přenosová média**
  + metalické kabely
    - UTP
    - STP
    - koaxiální kabel
  + optické kabely
    - jednovidové
    - mnohavidové
  + bezdrátová média
    - rádiové, mikrovlnné, satelitní, infračervené

## Mobilní technologie

* všechny současné sítě využívají **buňkový systém**
  + hodně malých vysílačů
  + každý si obhospodařuje svoje území
* umožňuje vícenásobné využití frekvenčního pásma
  + více účastníků v jedné síti může komunikovat ve stejném čase na jedné a téže frekvenci
* nejčastěji se využívá šestiúhelníkové uspořádání buněk
* **architektura sítě GSM**
  + **USS – Uživatelský subsystém** 
    - mobilní stanice
    - datová zařízení (modemy, pda…)
  + **BSS – Subsystém základnové stanice**
    - základnová stanice (BTS)
    - kontrolér základnových stanic - *Base station controller* (BSC)
  + **NSS – Subsystém sítě** 
    - ústředna (MSC)
    - bezpečnostní a identifikační registry – Autentifikační centrum - *Authentication center* (AuC)
    - Domovský lokační registr - *Home location register* (HLR)
    - Návštěvnický lokační registr - *Visitor location register* (VLR)
    - Identifikační registr – *Equipment Identification Register* (EIR)
* **GSM sítě**
  + *Global System for Mobile communication*
  + standard pro mobilní telefony
  + digitální komunikace
  + celosvětové pokrytí
  + **2G**
    - GSM900, GSM1800
    - 80. léta
    - standardizace, přenos hlasu
  + **2.5G**
    - podpora paketových přenosů pro datovou komunikaci GPRS (*General Packet Radio Services*)
    - datové přenosy
      * **HSCSD** (*High Speed Circuit Switched Data*)
      * **EDGE** (*Enhanced Data Rates for Global Evolution*)
  + **3G**
    - vysokorychlostní přenos hlasu i dat
    - telefony 3. generace od r. 1999
    - UMTS, CDMA
  + **3.5G**
    - datové přenosy o rychlosti 3Mbit/s
  + **4G**
    - nastupující standard, míněná rychlost alespoň 100Mbit/s, ale teoreticky možná i přes 1Gbit/s
    - zabezpečené rychlé spojení celého světa pro přenos hlasu, digitálního videa a dat vysokou rychlostí

# Ethernet

Principy fungování, vývoj a topologie, přístupová metoda, síťová karta, strukturovaná kabeláž

## Principy toku dat v LAN

* dnes se prakticky využívaný jen Ethernet a ATM
* ***Token******Ring***
  + topologie = kruh
  + stanice si počká, až k ní přijde token, změní 1→ 0 a přidá hlavičku a data, pak to pošle dál,

přijde to k adresátovi, ten to označí jako „rámec v pořádku přijat“ a pošle dál původní stanici jako potvrzení

* + po převzetí původní stanicí může tato stanice vysílat dál, nebo změnit 0→1 a poslat dál prázdný token
  + nevýhodou je **plýtvání časem**
    - každý token jde tam a pak zase k původní stanici jako potvrzení
    - proto existují různé varianty token ringu, které zrychlují síť tak, že posílají do sítě více tokenů najednou
      * max. tolik kolik je stanic - v každém úseku max 1 token v 1 okamžik
  + **parametry**
    - rychlost šíření v síti „Token Ring“ - 4Mbit/s (což na r.78 bylo dost!), dále 6 Mbit/s, 16 Mbit/s (současný běh více tokenů)
    - max. počet stanic: 250 v jednom okruhu (jinak použít bridge = nejlevnější; nebo router = drahé)
* ***ARCNET***
  + přístupová metoda
    - tzv. „Token Bus“
    - distribuovaná přístupová metoda
    - stanice si vytváří „logický kruh“, každá stanice zná adresu svého následníka a může tedy adresovat dalšího v „kruhu“
    - každá stanice čeká, než k ní přijde token
    - stanice nemají pevně uloženou adresu na síťovce (jako dnes nebo jako u „Token Ringu“),
      * fyzická adresa se nastavuje pomocí osmi switchů (dipy)
      * administrátor nastaví jedinečnou kombinaci (max. 255 stanic)
  + **parametry**
    - max. počet stanic 255 (0 – všeobecná)
    - rychlost 4-8 Mbit/s, ale i 20 Mbit/s
  + „Token Ring“ a „ARCNET“ jsou typy sítí, kde se „pořád něco děje“
    - pořád se vysílá (minimálně obíhá alespoň „pešek“)
* ***ETHERNET***
  + určen k nahrazení sítí „ARCNET“
* ***FDDI***
  + *FDDI = Fiber Data Distributed Interface*
  + vysokorychlostní síť
  + propustnost 100 Mbit/s
  + běží na optických vláknech, topologie zdvojený kruh (návrháři ideově čerpali ze sítě token ring)
  + jsou zde však vysoce inteligentní zařízení schopná detekovat závadu na trase
    - případně přejít na druhý (záložní) okruh
    - ten je použit jen pro zálohu (tzn. 50% redundance)
  + vysoká cena

## Ethernet

* technologie pro lokální počítačové sítě, založená na přenosech paketů
* definuje
  + vedení a jeho instalaci
  + formáty paketů
  + protokoly
  + metodu řízení přístupu (CSMA/CD).
* původně
  + navržena firmou Xerox Corporation
  + pracující rychlostí10 Mbit/s
  + s použitím přístupové metody CSMA/CD
  + na koaxiálním kabelu.
* Bob Metcalfe nazval novou technologii s všesměrovým vysíláním dle teorie z 19. století o neprostupujícím "etheru", kterým se šíří elektromagnetické vlny "Ether" net.
* název je v současnosti používán s odkazem na všechny LANsítě, které používají přístupovou metodu CSMA/CD.
* Ethernet byl původně navržen pro sítě se sporadickým provozem
* specifikace IEEE 802.3 byla odvozena z původní technologie Ethernetu v osmdesátých letech
* verze Ethernet 2.0, která byla vyvinuta firmami Digital Equipment Corporation, Intel Corporation a Xerox Corporation, je již plně kompatibilní s normou IEEE 802.3.
* Ethernet není síť jako taková, ale definice, předpis, standard
* ucelená představa o tom, jak má lokální síť fungovat
* v rámci vrstvových síťových modelů pokrývá jen tzv. fyzickou a linkovou vrstvu
* **první Ethernet** pracoval s přenosovou rychlostí 3 Mb/s (přesně: 2,94)
  + principy sítě Ethernet byly vyvinuty r. 1976 firmou Xerox
    - standardní Ethernet, pro komerční využití zhruba od roku 1980 – standard IEEE 802.3
  + určen k nahrazení sítí ARCNET
  + existuje přibližně dvacet variant sítí ethernetového typu.
  + nejsme schopni garantovat, kdy data přijdou (nehodí se moc pro video apod.)
  + rychlost přenosu 10 Mb/s – 1000 Mb/s
  + maximální vzdálenost mezi stanicemi 2,8 km
  + maximální počet stanic 1024
  + topologie – sběrnice
  + problém s topologií kruh
* **kabeláž**
  + pro standardní síť Ethernet je předepsán speciální kabel
    - šířka půl palce
    - impedance 50  2 Ohm (tlustý kabel)
    - vnitřní žíla postříbřený drát, čtyřnásobné opletení
    - oba volné konce musí být impedančně přizpůsobeny zakončovacími odpory 50 Ohm.
  + standardy IEEE 802.3 i koaxiální kabel (tenký Ethernet – koaxiál 0,2 palce)
  + (nestíněná) kroucená dvojlinka
  + optický kabel

## přístupová metoda

* **CSMA** 
  + *Carrier Sense Multiple Access*
  + stanice připravená vysílat data si "poslechne", zda přenosové médium (kabel) nepoužívá jiná stanice
  + v případě, že ano, stanice zkouší přístup později až do té doby dokud není médium volné
  + v okamžiku kdy se médium uvolní, začne stanice vysílat svá data
* **CD** 
  + *Collision Detection*
  + stanice během vysílání sleduje, zda je na médiu signál odpovídající vysílaným úrovním (tedy aby se např. v okamžiku kdy vysílá signál 0, nevyskytl signál 1)
  + případ, kdy dojde k interakci signálů více stanic, se nazývá kolize
  + v případě detekce kolize stanice generuje signál *JAM* a obě (všechny) stanice, které v daném okamžiku vysílaly, generují náhodnou hodnotu času, po níž se pokusí vysílání zopakovat.
* **CSMA/CD** v základním pásmu
  + přenos v základním pásmu znamená, že po kabelech proudí vždy jen jeden logický signál
  + stanice, která chce vyslat data, testuje stav kanálu
  + je-li kanál volný, zahájí vysílání, přitom však musí zaručit dodržení mezirámcové mezery 9,6 mikrosekund
  + je-li kanál obsazen, čeká na jeho uvolnění, poté ihned po uplynutí mezirámcové mezery zahájí vysílání
  + během přenosu testuje souhlas signálu na přenosovém médiu s vysílanými daty
  + je-li během celé doby vysílání signál správný, je považováno na této úrovni vysílání dat za úspěšně ukončené
  + zjistí-li však stanice nesouhlas, znamená to, že došlo ke kolizi se současným vysíláním další stanice
    - ihned přeruší přenos rámce
    - vyšle speciální rušící signál (jamming signal), aby kolizi bezpečně rozeznaly i ostatní zúčastněné stanice
  + po odeslání rušícího signálu čeká stanice po určitou dobu (backoff) a poté začne krokem 1 opakované vysílání
  + doba čekání se stanoví jako násobek doby potřebné k odeslání 512 bitů s náhodným číslem
  + náhodné stanovování doby čekání po zjištěné kolizi je nutné proto, aby opakované vysílání nezahájily stanice zúčastněné na kolizi ve stejnou dobu
  + pokud rámec není odeslán ani po 16 pokusech, je hlášena chyba
* **popis rámců**
  + **formát rámce**
    - rámec předávaný po síti Ethernet je sestavován v podvrstvě MAC (Medium Access Control) linkové vrstvy.
  + **synchronizační pole**
    - *preamble*
    - poskytuje signál, který umožní zasynchronizování generátoru hodin přijímací strany
    - nenese žádnou informaci a je k rámci připojeno ve vysílacích obvodech
    - do celkové délky rámce se nezahrnuje
  + **cílová adresa**
    - *Destination Address*
    - obsahuje adresu nastavenou na desce adapteru sítě stanice, které je rámec určen
    - může být individuální fyzická nebo skupinová
    - zvláštní postavení má všeobecná adresa (broadcast)
      * na ní vyslané rámce jsou přijaty na všech stanicích v síti
    - 48 bitů
  + **zdrojová adresa**
    - *Source Adress*
    - uvádí fyzickou individuální adresu desky připojení sítě, která rámec vyslala
  + **typ zprávy**
    - je na této úrovni v podstatě nevyužit,
    - jediný definovaný typ je určen pro diagnostické testování
  + **data**
    - mají délku 46 až 1500 slabik
    - minimální délka je stanovena proto, aby byl zaručen čas vysílání potřebný pro bezpečnou detekci případné kolize
  + **kontrolní pole** **rámce**
    - FCS (Frame Check Status)
    - označení obsahu rámce hodnotou CRC (Cyclic Redundancy Check)
    - při příjmu je hodnota CRC kontrolována a rámce, u kterých je zjištěn nesouhlas, jsou označeny za chybné
* **formát paketu** 
  + všechny rychlostní modifikace Ethernetu používají stejnou komunikační metodu CSMA/CD
  + používají však i stejný formát a velikost paketu
  + ethernetový paket je definován na 1. a 2. vrstvě OSI
  + základní částí paketu je hlavička linkové vrstvy, která je následována daty (včetně hlaviček vyšších vrstev)
  + hlavičky jsou principielně 4 typů a jsou vzájemně nekompatibilní:
    - *Ethernet\_II*
    - *Ethernet\_802.3*
    - *Ethernet\_802.2*
    - *Ethernet\_SNAP*
* každý paket je uvozen preambulí
  + slouží k synchronizaci vysílající stanice a přijímajících stanic
* následuje
  + adresa určení (MAC)
  + zdrojová adresa (MAC)
  + číslo označující typ paketu
    - typ paketu obsahuje číslo větší než 0x05DC
    - jako příklad může být použit např. číslo 0800, označují IP paket nebo 8137 označují Novell IPX paket
    - ostatní čísla lze najít např. v RFC např. 1700.
  + datová část
  + kontrolní součet

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Preamble | adresa určení (DA) | zdrojová adresa (SA) | typ paketu | data | CRC |
| 8 byte | 6 byte | 6 byte | 2 byte | 46 až 1500 byte | 4 byte |

## Síťová karta

* *NIC (Network Interface Card)*
* fyzické rozhraní mezi počítačem a sítí
* připravuje data z počítače na přenos po fyzické vrstvě
* posílá data do sítě, přijímá data ze sítě
* kontroluje tok dat mezi PC a sítí
* **obsahuje**
  + hardware
  + firmware
    - řídí činnost hardware
    - obsluhuje *LLC* a *MAC* podvrstvu linkové vrstvy modelu *OSI*
* **rychlost**
  + poloduplexní (vysílání nebo příjem)
  + plně duplexní (vysílání i příjem)
* **sběrnice**
  + *PCI*
  + integrovaná na MB
    - *WakeOnLan*

## Síťové prvky

* ***Repeater***
  + ke spojování ethernetových segmentů
  + zesiluje signál, upravuje hrany
  + zabývá se jednotlivými bity
  + umožňuje prodloužit maximální délku kabelu
* ***Switch***
  + přepíná rámce
  + přenos probíhá jen mezi příslušnými porty komunikujících stanic, tzv. přepínaný ethernet
  + dle cílové MAC adresy zajišťuje
    - přepnutí
    - zrušení (zdroj=cíl)
    - přesun na vyšší síť (není-li adresa v tabulce)
  + *Store and Forward*
    - načte celý rámec a pak s ním pracuje
  + *Cut through*
    - načte jen záhlaví a podle něj rámce pošle dál
* ***Router***
  + rozhoduje o tom, kam bude paket odeslán, aby se dostal k cíli optimální cestou
  + pracuje na 3. vrstvě OSI
  + multiprotokolové routery
  + obsahují směrovací tabulku se sousedními sítěmi a adresami bran do nich

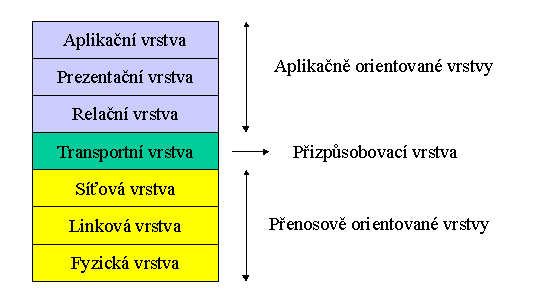
## Strukturovaná kabeláž

* **koaxiální kabel**
  + tenký (185m)
  + tlustý (500m)
* **kroucená dvoulinka**
  + nestíněná (UTP)
  + stíněná (STP)
  + nejčastěji Cat5 (100m)
  + **křížená**
    - *Cross-Over*
    - dva kabely jsou prohozené na jedné straně (RJ-45)
    - jedna strana slouží pro jednu stranu jako vysílač a pro druhou jako přijímač a opačně
    - při zapojení PC2PC
  + **nekřízená**
    - *Straight-Through*
    - při zapojení aktivní prvek 2 PC
* **optika**
  + jednovidové
  + mnohovidové
  + až stovky kilometrů
* **pasivní prvky**
  + zásuvky
  + kabely
  + patch panely
  + konektory
  + nástroje
* **aktivní prvky**
  + síťové karty
  + konvertory
  + hub, switch
  + router
* ***10 Base X***
  + 10 znamená rychlost v Mb/s
  + BASE znamená přenos v základním, nepřeloženém pásmu
  + *číslo* = maximální elektrická délku segmentu ve stovkách metrů
  + *písmeno* = typ kabelu
* **10–BASE 5**
  + 1. normovaná Ethernetová síť
  + každá připojená stanice znamená zkrácení maximální délky o přibližně pět metrů
    - to způsobují přechodové odpory na konektorech apod
  + používá se pětkrát stíněný koaxiální kabel ozn. jako Thick Ethernet neboli Yellow Cable s impedancí 50 ohm, (tlustý koaxiální kabel  tlustý Ethernet – průměr přes 1 cm),
    - signál se šíří rychlostí 0,71c (normálně v metalických vodičích 0,69c)
  + kabel fungoval podstatně lépe, ale náročné požadavky na zapojení kabeláže
    -  max. délka 500 m
  + připojování stanic pouze ve vyznačených místech transcieverem
    - pouze ve vzdálenostech násobku 2,5 metrů kabelu (na kabelech bývá označení) a max 50 m od transcieveru
  + segment musí být na obou koncích ukončen pomocí tzv. Terminátorů
  + nesměly se využít 2 sousední přípojné body
    -  minimální rozteč mezi stanicemi je 5 metrů
  + **nevýhody**
    - Cena
    - tlustý, neohebný kabel
    - speciální zámkové konektory (zámek v podobě závitu)
  + **konektory**
    - zámkové konektory
  + **topologie**
    - sběrnice
  + **přístupová metoda** 
    - CSMA/CD
  + **maximální počet stanic**
    - 1024
  + adresu tvoří číslo o délce 12 bytů, které je pro každý síťový adaptér unikátní
    - s vyjímkou několika typů, kde ho je možné definovat
  + přes repeater se mohou spojit maximálně tři segment
  + používá se páteřní struktura
    - multiport repeater – připojení více segmentů.
  + SWITCH slouží k rozdělení sítě na více Ethernetových sítí
    -  má paměť na 1024 adres, které mohou komunikovat jeho prostřednictvím
* **10-BASE T**
  + použití kabeláže Twisted Pair (kroucená dvoulinka kategorie 3, stíněná nebo nestíněná) s impedancí 100 ohm, T-twist  kroucená dvoulinka
  + topologie (strom, hvězda)
  + maximální vzdálenost mezi stanicemi je 420 metrů (počítá se i s přechodem přes HUB Pouze aktivní HUBy – pasivní jen u Arcnetu)
  + **konektory**
    - RJ-45
      * rear jack
      * u telefonů
      * kostička s packou nahoře nedovolí konektrou se vysunout
      * šířka 8 kontaktů – pinů
      * zapojují se max. 4 vodiče
      * 2 vodiče pro přenos tam a 2 pro přenos zpátky
* **10-BASE F**
  + F – fiber channel
  + norma, která umožňuje použití na optice
  + délka dána typem vlákna
  + vše plně propojitelné přes media konektory
  + v době objevení nejrychlejší síť (Ethernet)
* **10Base-FL**
  + používá jako přenosové médium multimodový optický kabel
  + délka kabelu mezi uzly může být max. 2 km
  + existuje i modifikace používající singlemodový optický kabel
* **100-BASE T**
  + na kabelech 10-BASE T, ale využito všech osm vodičů
  + rychlost 100 Mb/s
  + neujalo se
  + vzdálenost od zásuvky k datovému rozvaděči nesmí překročit 90 m
  + 1024 stanic
  + *CSMA / CD*
* **100 BASE TX**
  + vylepšený 100-BASE T
  + kabeláž kategorie 5
  + 10x rychlejší technologie
  + kabel až 210 metrů, od HUBu 90 metrů
  + větší ztráty na spojích
  + používá jako přenosové médium kroucený dvoudrát (stíněný nebo nestíněný) s impedancí 100 ohm (min. Cat 5)
* **100-BASE FX**
  + 100-BASE TX s využitím optických kabelů
  + délka kabelu mezi uzly může být v případě plně duplexního provozu max. 2 km
    - v příp. polovičního duplexu je vzdálenost ovlivněna zapojením sítě
  + X – propojení 10 a 100 Ethernetu: média konvertory
* **1000-BASE TX**
  + kabeláž kategorie 6, 210 m
* **1000-BASE FX**
  + délka 2 km (přes SWITCH můžeme táhnout dál)
* **1000BASE-SX** 
  + používá jako přenosové médium multimodový optický kabel
  + délka kabelu mezi uzlem a aktivním prvkem je ovlivněna parametry kabelu
* **1000BASE-LX** 
  + používá jako přenosové médium ultimodový nebo singlemodový optický kabel
  + délka kabelu mezi uzlem a aktivním prvkem je ovlivněna typem a parametry kabelu

# RM ISO/OSI, TCP/IP

Popis a srovnání, funkce zásadních protokolů, IP adresy

## Referenční model ISO OSI

* „Open System for Interchange of Information“
* **důvod vzniku**
  + v 70. letech každá firma vyráběla jiné systémy
    - potřeba jednotného základu (standardu)
  + koncepce ISO/OSI je "oficiální" koncepcí počítačových sítí, pocházející od mezinárodní organizace ISO
    - sdružuje národní standardizační (normalizační) organizace
* původně měl být vytvořen standard, který by pokrýval veškeré otázky kolem architektury tzv. otevřených systémů, ukázalo se však, že něco takového by bylo příliš náročné a zdlouhavé, a tak se autoři omezili pouze na problematiku vzájemného propojování uzlů sítě (chápaných jako "otevřené systémy")
* odsud také název OSI, který znamená: **Open Systems Interconnection**
  + doslova: propojování otevřených systémů
* i toto omezení však stále přinášelo příliš velký objem práce
* nakonec bylo rozhodnuto o řešení ve formě "referenčního modelu"
* bylo definováno
  + kolik má být vrstev
  + jaké má být rozhraní mezi jednotlivými vrstvami
  + co má která vrstva vykonávat a jaké služby má poskytovat
* ale již nebylo přesně specifikováno jak konkrétně má která vrstva své funkce vykonávat
  + nebyly připraveny jednotlivé protokoly
  + tyto konkrétní protokoly samozřejmě vznikaly a nadále vznikají, ale formálně už jako samostatné protokoly, pouze "logicky" zasazené do rámce referenčního modelu ISO/OSI
* celková koncepce ISO/OSI počítá celkem se **sedmi vrstvami**, které je možné rozdělit do **dvou hlavních skupin**:
  + **přenosové vrstvy**
    - tři nejspodnější
    - mají na starosti přenos dat
  + **vrstvy orientované aplikačně**
    - tři nejvyšší
    - mají za úkol poskytovat potřebnou podporu jednotlivým aplikacím
  + mezi těmito dvěma skupinami pak existuje ještě jedna vrstva
    - má za úkol vzájemně přizpůsobovat možnosti a potřeby obou hlavních skupin
    - 
* **vrstvy**
  + *hardwarově závislé*
    - závislé na daném HW
    - vyměníme-li HW, potřebujeme např. i nové ovladače
  + *hardwarově nezávislé*
  + *transportní skupina*
    - zajišťuje vlastní přenos informace mezi místy
    - z pohledu relační vrstvy se vše jeví, jako kdyby byly všechny počítače přímo spojeny dráty se všemi ostatními
  + *aplikační skupina*
    - připravuje data a je blíže k uživateli
* **fyzická vrstva**
  + přenáší fyzicky data z bodu A do B
  + zajišťuje zakódování rámců do podoby elektrických signálů a jejich přenos po síťových kabelech
  + data se z pohledu vrstvy jeví jako posloupnost bitů (zpracovaný signál) a nemá žádnou strukturu
  + umí jen poslat/přijmout 0 či 1 (bit)
  + definuje
    - způsob signálu (elektrický signál, světelný paprsek, zvuková vlna…) a jeho limity (minimální/maximální úroveň signálu, napětí signálu…)
    - typ přenosového média (skleněné vlákno, vzduch, vodič…)
    - dálková omezení, kvalitativní omezení…
    - topologie
      * způsob propojení mezi sebou (jak jsou dráty propojeny mezi sebou)
* **linková vrstva**
  + řeší 2 hlavní věci
    - fyzickou adresaci
    - použitou přístupovou metodu
  + pakety jsou zde přetvářeny na **rámce** (frames)
    - ty jsou vybaveny navíc příslušnými adresami a posloupnostmi dat, které jsou potřebné pro synchronizaci hodin síťových adaptérů na spolupracujících stanicích
* **síťová vrstva**
  + řeší logickou adresaci, tzn. výměna dat mezi oddělenými (různými) sítěmi
  + vybírá optimální cestu pro přenos zprávy k cílové stanici
  + u jednoduchých lokálních sítí, kde mají všechny síťové stanice společnou kabeláž a nejsou připojeny na jinou lokální síť, nemusí tato vrstva být přítomná
* **transportní vrstva**
  + dohlíží na veškeré přenosy dat
  + dojde-li k jejich ztrátě, vyžádá je nanovo → zaštiťuje celý přenos
  + z pohledu 5. vrstvy (relační) zajišťuje, aby byla síť virtuálně propojená „každý s každým“
  + má na starosti ochranu přenášených dat
  + data jsou zde rozdělována a slučována do menších částí tzv. paketů (packets)
  + zde jsou prováděny kontrolní součty, které umožňují následnou kontrolu správnosti přenosu
  + také se zde vytvářejí záložní kopie, pro případné opakování přenosu pokud je neúspěšný
  + opět je ke zprávě připojena hlavička této vrstvy
* **relační vrstva**
  + řídí relace - začíná je a končí
  + definuje základní bezpečnostní systém – přístupová práva
  + nazýváme ji také „auditní“ (umožňuje monitorovat činnost)
  + zajišťuje zabezpečení vlastního spojení mezi síťovými stanicemi (řízení komunikace, náhlé přerušení komunikace)
  + v této vrstvě se rozhoduje také o tom, jakým způsobem bude spojení (relace) realizováno
  + spojení mezi počítači lze realizovat buď
    - **poloduplexně** (počítače se střídají v příjmu a vysílání)
    - **duplexně** (počítače zároveň vysílají a přijímají)
* **prezentační vrstva**
  + zajišťuje prezentaci dat, tzn.:
    - **kódování** (způsob uložení informace - uložení informace v nějakém jazyce; např. kódování čísel; změna kódování je i jednoduchou „šifrou“)
    - **šifrování** (skrytí informace, aby nebyla od relační vrstvy níž přístupná)
    - **komprese** (stlačení dat na menší objem - zvyšuje množství přenášených dat za časovou jednotku - např. užití binárních stromů)
* **aplikační vrstva**
  + obsahuje aplikační protokoly, jejichž prostřednictvím komunikuje aplikace s OSI modelem
  + např. protokoly SMTP, FTP, HTTP + služby k jejich použití
  + jediná síťová vrstva, ke které má uživatel přímý přístup
  + v této vrstvě jsou data, která chceme poslat na jiný počítač, převedena za pomoci aplikací do počítačové podoby – k takovéto zprávě je přiložena tzv. hlavička aplikační vrstvy, ve které je obsažena informace identifikující vysílající a cílový počítač

## IP adresy

* **rozdělení **
  + každé rozhraní má svou adresu
  + 32bitové číslo (4 bajty)
  + tečkovaný desítkový zápis (například 147.230.16.8)
  + celosvětově jednoznačné
  + distribuované přidělování
* když se autoři protokolů TCP/IP rozhodovali, jaké adresy zvolit pro síťovou vrstvu, mohli použít jako základ fyzické adresy (adresy používané na úrovni vrstvy síťového rozhraní)
* přesně tak to učinili například autoři protokolů IPX/SPX od firmy Novell
* výhodou tohoto řešení je snadný převod síťových adres na fyzické
* autoři protokolů TCP/IP se ale rozhodli pro jiné řešení
  + zvolili zcela nové a ryze abstraktní adresy (abstraktní v tom smyslu, že nemají přímý ekvivalent v žádných fyzických adresách)
  + pro jejich rozsah zvolili určitý kompromis mezi největšími fyzickými adresami (což jsou zřejmě 48bitové ethernetové) a jejich nejmenšími variantami - rozhodli se pro adresy v rozsahu 32 bitů
* **vnitřní struktura IP adres**
  + na každou jednotlivou IP adresu je možné se dívat jako na souvislý řetězec 32 bitů, neboli jako na jedno 32bitové binární číslo IP, bez jakéhokoli vnitřní struktury (tj. bez toho, aby jednotlivým skupinám bitů v rámci této adresy byl přisuzován specifický význam)
  + pro některé účely (například pro potřeby směrování) je ale přece jen výhodnější dát IP adresám určitou vnitřní strukturu
    - konkrétně možnost jejich rozčlenění do dvou logických složek, které po řadě odpovídají **adrese (číslu) sítě** jako takové a **relativní adrese (číslu) konkrétního uzlu** v rámci této sítě.
* **tři třídy IP adres**
  + důvod zavedení tříd
    - aby se neplýtvalo s omezeným počtem IP adres v malých sítích a naopak aby velké sítě neměly nedostatek IP adres pro své počítače
  + autoři protokolů TCP/IP od začátku předpokládali, že v praxi se budou vyskytovat jak sítě hodně velké, neboli sítě s velkými počty uzlů (kterých ale nejspíše bude málo), tak i sítě s malými počty uzlů (kterých ale může být hodně) a samozřejmě i sítě středně velké
  + tomuto předpokladu pak přizpůsobili konkrétní způsob rozdělení 32bitových IP adres na dvě logické části, který nefixovali způsobem naznačeným v předchozím odstavci
  + místo toho navrhli tři "předdefinované" způsoby rozdělení IP adres na obě logické části, odpovídající očekávaným potřebám sítí velkých, středních a malých
  + tímto způsobem vznikly tři třídy adres, označované po řadě jako **A, B a C**
* **ve třídě A**
  + je pomyslná dělicí čára vedena mezi prvním a druhým bytem celé čtyřbytové adresy
  + to znamená, že pro relativní adresu uzlu v rámci sítě jsou určeny plné 3 byty (tj. 2 na 24 bitů)
  + pro vyjádření síťové části adresy slouží pouhých 7 bitů (a 8. bit v prvním bytu slouží k rozlišení tohoto druhu adresy)
  + třída A počítá s existencí až 2 na 7 (neboli 128) opravdu velkých sítí, z nichž každá může mít až 2 na 24 uzlů
* **třída B**
  + počítá s tím, že pomyslná dělicí čára vede mezi druhým a třetím bytem celé čtyřbytové adresy
  + to znamená, že pro síťovou část adresy je vyhrazeno 14 bitů (dva zbývající do šestnácti jsou nutné pro identifikaci tohoto typu adresy)
  + pro relativní část adresy je k dispozici 16 bitů
  + adresy třídy B tedy počítají s 2 na 14 různých sítí, z nichž každá může mít nejvýše 65536 uzlů
  + jde tedy o "střední" sítě.
* **v případě třídy C**
  + pomyslná dělicí čára vedena mezi třetím a čtvrtým bytem, takže pro relativní část adresy zbývá jediný byte
  + adresy tohoto typu jsou tedy určeny pro malé sítě s maximálně 256 uzly
  + zato však takovýchto sítí může být velký počet, protože pro síťovou část adresy je vyhrazeno 21 bitů (24 bez tří identifikačních bitů)
* existující dvě další třídy IP adres
  + **jsou to adresy třídy D**
    - určené pro tzv. multicast (skupinové vysílání)
  + **skupiny E** (pro budoucí využití)
  + adresy těchto tříd jsou zajímavé tím, že jsou logicky jednosložkové
* **speciální adresy**
  + **127.x.x.x**
    - loopback (neopouští počítač) pro test síťového stacku
  + **x.255.255.255**
    - síťový broadcast (paket je vyslán na všechny počítače, ale pouze v síti)
  + **255.255.255.255**
    - všeobecný broadcast
    - paket je vyslán na všechny počítače i mimo síť
    - aby neobíhal donekonečna, je mu přiřazen „time to live“ (TTL)
      * každý router sníží jeho hodnotu o jedna a ve chvíli, kdy TTL = 0 paket příslušný router zahodí

## Protokol

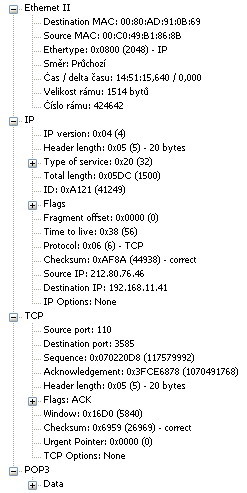
* protokol je obecně sada formálních pravidel definujících přenos sítí (obsah, formát, čas…)
* zajišťuje výkon služby v dané vrstvě
* spolupracuje s dalšími protokoly téže vrstvy a vrstev sousedních
* komunikace mezi stejnými vrstvami systémů je řízena
  + komunikačním protokolem za využití spojení, které vytvoří sousední nižší vrstva

## IP protokol

* IP = Internet Protocol (RFC 791)
* bez spojení (samostatné datagramy)
* bez záruk (best effort)
* zabezpečuje správné doručování dat jednotlivým počítačům v síti
* **pracuje na úrovní síťové vrstvy**
* nespolehlivý (nenavazuje spojení – časté chyby)
* každý datagram představuje samostatnou jednotku
  + putují sítí nezávisle
  + pořadí doručení nemusí být stejné jako pořadí ve zprávě
  + doručení není zaručeno, spolehlivost zajišťují vyšší vrstvy (TCP)
* **hlavička**
  + IP adresu zdroje
  + IP adresu cíle
  + verze (v současnosti verze 4, přechází se pomalu na v6)
  + délka hlavičky
    - v 32bitových slovech (max. 60 B)
  + *TOS*
    - *Type of Service* (požadavky na přepravu)
  + celková délka
    - max. 65535 B
  + *TTL*
    - *Time to Live*
    - každý směrovač zmenší alespoň o 1, vynuluje-li se, zahodí paket a pošle odesilateli ICMP zprávu
    - ochrana proti zacyklení
  + CRC
    - nezahrnuje data

## 

## TCP protokol

* *TCP = Transmission Control Protocol*
* spolehlivá přeprava (vyžaduje řada aplikací)
* proud bitů bez struktury (bitová roura)
* spojovaná služba, virtuální okruhy
* vyrovnávací paměti – komunikační software seskupuje/rozděluje data podle potřeby, aby komunikace byla efektivní
* **pracuje na úrovni transportní vrstvy**
* plně duplexní spojení
* doručí všechna data bez ztráty a ve správném pořadí
* jde o službu se spojením
  + má tedy fáze
    - navázání spojení
    - přenos dat
    - ukončení spojení
* rozlišuje aplikace pomocí portů
* **hlavička**
  + čísla portů
  + délka hlavičky v 32bitových slovech
  + příznaky
    - URG = segment obsahuje urgentní data
    - ACK = obsahuje potvrzení
    - PSH = předat data cílové aplikaci co nejrychleji
    - RST = reset (náhlé ukončení spojení)
    - SYN = zahájení spojení (synchronizace pořadových čísel)
    - FIN = končím odesílání dat (polouzavření)
  + kontrolní součet
    - pseudohlavička + TCP hlavička + data
* **zajištění spolehlivosti**
  + potvrzování řeší ztráty paketů
  + možnost přehození a duplikace
    - pořadová čísla
    - TCP čísluje bajty (oktety)
  + potvrzuje nejdelší souvislý prefix, který přijal od začátku vysílání (posílá číslo bajtu, který očekává)
  + jednoduché a jednoznačné
  + ztracené potvrzení nemusí způsobit opakování
  + ****nelze oznámit mezeru

## UDP protokol

* nespojovaný
* nespolehlivý, pouze s detekcí chyb
* nahrazuje TCP (jednodušší verze protokolu TCP)
* je proto vhodný pro aplikace, které přenášejí malá kvanta nezávislých dat

## ARP protokol

* slouží k mapování IP adres na MAC adresy
* spravuje paměť cache, umožňuje zjistit HW adresu všech počítačů v dané síti

## ICMP protokol

* nespojovaný
* nespolehlivý, pouze s detekcí chyb (kontrolní součet)
* **použití**
  + předávání chybových a diagnostických informací mezi stanicemi
  + ping

# Internet

Organizační struktura, vývoj, RFC dokumenty, domény, technické předpoklady pro připojení, hrozby

## Internet

* internet je celosvětový systém navzájem propojených počítačových sítí („síť sítí“), ve kterých mezi sebou počítače komunikují pomocí rodiny protokolů TCP/IP
* společným cílem všech lidí využívajících Internet je bezproblémová komunikace (výměna dat)
* Internet jsou volně propojené počítačové sítě, které spojují jeho jednotlivé síťové uzly
* uzlem může být počítač, ale i specializované zařízení (například router)
* každý počítač připojený k internetu má v rámci rodiny protokolů TCP/IP svoji IP adresu
* pro snadnější zapamatování se místo IP adres používají doménová jména
* největší světová **decentralizovaná** počítačová síť
* společenství lidí, kteří ho vytváří, využívají, a kteří spolu komunikují
* dnes velmi komerční, dříve sloužil jen vzdělávacím a vědeckým společnostem
  + obrovské rozšíření mezi řadové uživatele
* dnes řada negativních stránek
  + trestná činnost provozovaná prostřednictvím Internetu

## RFC

* RFC je zkratka anglického výrazu *request for comments* (žádost o komentáře)
  + používá se pro označení řady standardů a dalších dokumentů popisujících Internetové protokoly, systémy apod
* jak už název napovídá, RFC jsou oficiálně považovány spíše za doporučení než normy v tradičním smyslu, přesto se podle nich řídí drtivá většina Internetu
* jednotlivé RFC dokumenty vydává editor RFC podle příkazů ***Internet Architecture Board***
* každé RFC má při zveřejnění přiděleno číslo
* žádné jednou vydané RFC se nikdy neruší, pouze se v budoucnu může upravit vydáním novějšího RFC
* všechna RFC lze volně získat na adrese http://www.ietf.org/rfc.html a na mnoha dalších místech
* každé RFC je dostupné v podobě čistého ASCII textu (v angličtině), ze kterého se tvoří i další verze (např. HTML, PDF apod.)
* na rozdíl od klasických norem a standardů vydávaných klasickými normotvornými institucemi (jako např. ISO, ANSI apod.) vznikají RFC poněkud jiným způsobem
* původními autory jednotlivých RFC jsou obvykle konkrétní experti, kteří se snaží řešit konkrétní problém, jehož řešení nabídnou ve formě návrhu RFC Internetové veřejnosti (jako tzv. Internet Draft)
* pokud je dané řešení (často již dobře fungující v rámci nějakého pilotního provozu) uznáno za přínosné, dokument se vydá jako RFC
* toto pragmatické řešení standardů sestavovaných jednotlivci či malými skupinami na základě praktických zkušeností má mnohé výhody oproti formálnějším procesům standardizačních komisí u úřadů typu ISO
* standardy vytvořené pomocí RFC jsou kupodivu (vzhledem k neexistenci jakékoli skutečné moci na jejich vynucování) až na výjimky dodržovány, přičemž pomohly rozšíření Internetu do dnešních celosvětových rozměrů

## Vznik a historie sítě Internet

* zhruba před dvaceti roky došlo ve světě k obrovskému rozšíření osobních počítačů
* začaly vznikat počítačové sítě, světová síť umožnila počítačům „dospět“ a v neposlední řadě docílila toho, že informační zdroje, které poskytuje, se již netýkají pouze několika vynikajících odborníků, ale jsou běžnou realitou
* za vznik a úplné počátky sítě Internet se pokládá období kolem roku 1969
* v té době vzniká v USA experimentální síť ***ARPAnet***
  + předchůdce sítě Internet
  + tato síť byla vybudována pro účely podpory vojenského výzkumu
  + aby počítač mohl poslat zprávu, musel ji zabalit do obálky síťového protokolu Internet (IP) a tyto balíčky - pakety správně adresovat na cílový počítač
  + filozofie sítě spočívala v tom, že každý počítač v síti mohl komunikovat rovnocenně s kterýmkoliv jiným počítačem
  + snahou konstruktérů sítě *ARPAnet* bylo vybudování datové sítě mezi **Pentagonem** a dalšími strategicky důležitými institucemi
    - aby odolávala částečným výpadkům sítě (např. výpadkům z důvodu zničení části sítě při bombovém útoku, či jiné likvidaci)
    - klasická síť pro tyto účely není vhodná, protože tam, kde existuje řídící počítač (server) a ostatní stanice se k němu připojují jako terminály, není těžké zničit právě tento server a celá síť není schopna pracovat
    - proto tedy byla vyvinuta taková architektura, kde by fakticky neexistoval žádný centrální server a všechny počítače v síti by mohly komunikovat mezi sebou jako rovný s rovným
    - vznikla tak důmyslně propojená síť autonomních serverů, resp. lokálních sítí mezi sebou
  + zhruba od roku 1983, kdy se začaly lokální sítě velmi rychle šířit, začaly organizace dávat přednost připojení do sítě *ARPAnet*
    - namísto připojování svých lokálních sítí k velkým počítačům
    - aby tyto sítě a jejich uživatelé mohli komunikovat mezi sebou
* velkou roli v rozvoji sítě Internet sehrála síť ***NSFNET***
  + vznikla koncem 80. let
  + za vznikem stála **Národní vědecká nadace USA (NSF – National Science Foundation)**
  + síť tvořilo pět center na amerických universitách, vybavených superpočítači
  + tato síť vytvořila velký potenciál informací, který byl přístupný pro jakýkoli vědecký výzkum
    - nejen výzkum zbraní, jako tomu bylo u sítě ARPAnet
  + na architektuře sítě ARPAnet
  + tvořena vysokokapacitními telekomunikačními linkami
  + všechna střediska se superpočítači byla tedy propojena pomocí telefonních linek
  + protože by bylo finančně neúnosné pronajímat si vždy celou linku (až na určitou univerzitu k superpočítači), bylo rozhodnuto, vytvořit oblastní sítě
    - k páteři sítě by se připojovaly vždy přes svého nejbližšího souseda
    - jednotlivé řetězce potom byly připojeny v jednom bodě k superpočítači
    - vznikla tak síť lokálních sítí, která umožňovala každému jednotlivému uživateli (počítači) připojit se k páteři celé hlavní sítě a zároveň komunikovat s kterýmkoli jiným počítačem, který byl do této sítě sítí připojen
    - výslednými náklady potom byly pouze poplatky za připojení k nejbližšímu sousedovi
  + toto vše usnadnilo velký rozvoj Internetu mezi širší veřejnost
  + postupem času a s rostoucím množstvím nových klientů se síť stávala stále více přetížená, až se tato situace stala neúnosnou
  + stávající rychlost telefonních linek (56 000 bitů/s) nemohla stačit pro přenos

## Organizace

* **organizační struktura zajištění provozu sítě Internet**
  + strukturu ve smyslu organizačního zajištění provozu sítě a orgánů plní několik klíčových organizací
    - tyto organizace jsou téměř všechny dobrovolné
    - síť Internet jako celek nepatří nikomu
  + první takovou organizací je ***ISOC (Internet Society)***
    - ISOC je dobrovolná organizace,
    - hlavním cílem je podpora výměny informací pomocí technologie Internet
    - nyní pomáhá rozvojovým zemím s poradenstvím v oblasti sítí a připojení k Internetu
  + dalším orgánem **je *IAB (Internet Architecture Board)***
    - pracuje jako seskupení dobrovolníků
    - má za úkol řízení technické správy sítě Internet a stanovení standardů, týkajících se komunikace v síti
    - členové této organizace jsou jmenováni právě ISOC
  + ***IANA***
    - globální přidělování IP adres
    - spravuje TLD
  + ***IETF***
    - spravuje a rozvíjí Internetové komunikační standard
  + ***W3C***
    - spravuje a rozvíjí standardy pro WWW

## Připojení

* mezinárodní dálkové spoje dosahují v Internetu velmi vysokých přenosových rychlostí
  + tyto vysokorychlostní spoje nedosahují až ke koncovým uživatelům
* ti jsou k Internetu připojeni prostřednictvím tzv. „poslední míle“
  + samotné připojení uživatelů je realizováno různými technologiemi
  + uživatelé se někdy spojují do skupin, aby ušetřili náklady nebo naopak dosáhli na dražší a rychlejší připojení
  + zprostředkovatele připojení k Internetu označujeme ***Internet service provider (ISP)***
* v současnosti existuje několik možností pro **připojení počítače k Internetu**:
  + telefonní linka (majitelem linky je telefonní operátor)
    - využívá se modem
    - dříve se používalo vytáčené připojení, později ISDN a dnes různé varianty DSL
    - někdy je linka vyhrazena pouze pro datové přenosy
  + přípojka kabelové televize
  + bezdrátová datová síť
  + satelitní síť
  + mobilní telefonní síť
  + Wi-Fi
  + pomocí elektrické rozvodné sítě
* **o kvalitě připojení rozhoduje**
  + agregace
    - tj. kolik uživatelů sdílí jednu linku
  + doba odezvy
    - dlouhé odezvy mohou mít negativní vliv např. při internetové telefonii
  + rychlost připojení poslední míle
  + technologie použitá pro připojení poslední míle

## struktura

* Internet je síť vytvořená pospojováním lokálních sítí
* v síti Internet se můžeme setkat s nejrůznějšími druhy sítí
* není a ani nemůže být zajištěno, aby všechny LAN byly totožné a kompatibilní s připojením k Internetu (ke svému nejbližšímu poskytovateli)
  + je třeba kompatibilitu tohoto připojení nějak zajistit
  + k tomu slouží **ROUTERY** (směrovače)
    - router se nachází vždy někde na lokální síti a zprostředkovává připojení k Internetu
    - router od sebe odděluje lokální síť a Internet a chrání obě dvě části mezi sebou před negativními jevy buď uvnitř lokální sítě či Internetu
    - lokální síť je tedy chráněna před silným síťovým provozem, který je nasměrován do jiné oblasti a naopak Internet je chráněn před výpadky lokální sítě
    - zároveň je směrovač zařízením, které rozhoduje o tom, jakou další cestou (dále po Internetu) budou došlá data (pakety) odeslána
    - směrovač dané lokální sítě tedy nemá přímé spojení se všemi ostatními směrovači
    - zjistí si tedy adresu, kam jsou data zasílána, vybere nejvýhodnější cestu (další směrovač) a tou je odešle dál
    - tímto způsobem se data dostanou (podle uvedené adresy) až do určené lokální sítě k danému lokálnímu počítači

## Protokoly

* k tomu, aby byla vůbec možná komunikace mezi vzdálenými počítači v síti (a to platí pro jakoukoli síť, nejen pro Internet), je třeba dodržovat určitá pravidla
  + definují způsob zasílání a příjmu zpráv
  + k tomu slouží tzv. protokoly sítě
* protokoly jsou pravidla, podle kterých počítače na Internetu komunikují
* tyto protokoly mají za úkol vše, co se týká doručení zprávy z místa na místo
* každá zpráva, zasílaná prostřednictvím Internetu prochází nejméně třemi úrovněmi protokolů
* tyto protokoly jsou tedy jeden v druhém (jako ve vrstvách). Jsou to:
  + síťový protokol
  + transportní protokol
  + aplikační protokol
* ***Internet Protokol* - IP**
  + použití na Internetu jako síťový protokol
  + řídí a dohlíží na zasílání zpráv z lokální na cílovou stanici
  + stará se např.
    - o identifikaci příjemce a odesílatele (adresy)
    - o to jestli směrovače vědí, co mají udělat s došlými daty
  + je možné přirovnat k obálce s adresou odesilatele a příjemce, do které se vkládají data určená k odeslání.
  + pracuje s malými částmi zpráv – pakety
    - především z praktických důvodů, týkajících se efektivního využití sítě
    - obsahují obvykle 1 - 1500 bajtů,
    - celkovou zprávu musíme rozdělit do takto velkých segmentů a ty zasílat zvlášť
* ***Transmission Control Protokol***
  + TCP (protokol řízení přenosu)
  + pracuje na transportní vrstvě RM-OSI
  + spojová služba
  + spolehlivá služba
  + rozdělováním zpráv na pakety však nastávají problémy na přijímací straně,
    - je nutné tyto segmenty správně poskládat
    - o to se stará protokol TCP
    - Ten je transportním protokolem a obecně zajišťuje celistvost zprávy na přijímací straně.
  + na straně odesílatele má TCP za úkol:
    - rozdělit zprávu, určenou k odeslání na pakety
    - očíslovat pakety
    - zaslat po síti ve své vlastní TCP „obálce“, která je vložena do IP obálky, informace o očíslování paketů (pořadí), společně s částí přenášených dat
  + na straně příjemce má TCP za úkol:
    - „posbírat“ všechny TCP obálky
    - získat z nich data a tyto části složit ve správném pořadí do výsledné zprávy
    - požádat o opětovné zaslání paketu, pokud některý z nich po přenosu chybí či je deformován (kontrolu deformace provádí pomocí porovnávání kontrolních součtů), a zapouzdřit paket v TCP
* ***UDP (user datagram protokol)***
  + další přenosový protokol, který některé aplikace používají místo protokolu TCP
  + důvodem je fakt, že sestavení spojení a přenos pomocí TCP na jedné straně zajišťuje velkou pravděpodobnost, že vyslaná data ve správné podobě k adresátovi dojdou, na druhé straně však dochází k určitému zpoždění
  + toto zpoždění nemusí být pro určité aplikace žádoucí
    - proto používají UDP
  + UDP používá svou vlastní obálku UDP, která je vložena do IP místo obálky TCP
  + UDP je mnohem jednodušší než TCP a určité funkce z TCP vypouští
  + nezajišťuje například žádosti na zpětné vyslání poškozeného paketu a nestará se ani o sled paketů
  + v zásadě se UDP používá v takových aplikacích, které posílají jen krátké zprávy
    - které se svým rozsahem vejdou do jednoho paketu
    - a jsou schopné odeslat v případě ztráty zprávy paket znovu
  + o znovuvyslání poškozeného nebo nedošlého paketu se tedy stará samotná aplikace

## domény

* podnětem k vytvoření systému pojmenování domén - **DNS (Domain Name System)**
  + nebyla jen snaha o jednodušší práci
  + také nutnost zajistit jednoznačnou identifikaci rychle přibývajícímu množství stanic připojených k Internetu
* organizace v DNS je taková, že IP adrese je přiděleno jméno, které je reprezentováno soustavou tzv. domén
* jednotlivé domény se oddělují tečkami a v rámci jména se jich může vyskytnout prakticky libovolný počet
* zpravidla jich však není více než pět
* v doméně je úplně vlevo uvedeno jméno hostitelského počítače a za ním následují jména domén, do kterých počítač náleží.
* existují tzv. Organizační domény nejvyšší úrovně, které vznikly již s vytvořením DNS
  + existuje jich celkem sedm a vypovídají o druhu organizace, kde se počítač s touto doménou nachází
* s rozvojem Internetu a pronikáním této sítě do států celého světa bylo rozhodnuto, že odpovědnost za přidělování a organizaci jednotlivých jmen bude ponechána jednotlivým zemím
* vznikl tedy seznam domén, skládajících se ze dvou písmen, která odpovídají nejvyšším úrovním domén pro jednotlivé země
* doménové jméno se vyhodnocuje zprava
  + postupně se zpřesňuje

# Směrování

Základní principy, směrovací protokoly, směrovací algoritmy, směrovače

## Základní principy

* směrování (routování) je základní funkcí síťové vrstvy
* provádí se za **pomocí směrovací tabulky**
* routovací protokoly tvoří inteligenci sítě
* router vidí díky tabulce dále než ke svému sousedovi
* aktualizace směrovacích tabulek má povahu řídící zprávy – vyšší priorita
* vlivem zpoždění při přenosu tabulek mohou vznikat routovací smyčky
* pakety jsou podle toho směrovány jedním ze základních způsobů routování
  + **statické** (neadaptivní):
    - ručně zadané cesty (záznamy v routovací tabulce)
    - bezpečné a dobré
    - nereflektuje změny v topologii sítě
  + **dynamické** (adaptivní):
    - síť se automaticky přizpůsobuje změnám v topologii a dopravě
    - automaticky se vypočítávají cesty pomocí routovacího protokolu
    - dvou základních typů
      * ***Distance-Vector Routing Protocol***
      * ***Link-State Routing Protocol***
    - dále dělíme podle toho, zda jsou určeny pro nasazení uvnitř lokální sítě (přesněji řečeno uvnitř autonomního systému (AS), který může obsahovat několik LAN) nebo fungují napříč sítěmi (spojují AS dohromady)
      * ***Interior Gateway Protocol - IGP*** - routuje uvnitř Autonomous System (AS)
      * ***Exterior Gateway Protocol  - EGP*** - routuje mezi AS
* ***Distance-Vector Routing Protocol***
  + *RIP, RIP2, IGRP, EIGRP, BGP*
  + routery udržují routovací tabulku s informací o (vektoru) vzdálenosti do dané sítě
  + periodicky routovací tabulku zasílají sousedům, ti si upraví svoji tabulku a tu opět odešlou dál
  + pro výpočet nejlepší cesty se používá jedna (počet hopů u RIP) nebo více (propustnost linky a zpoždění u IGRP) metrik
  + jednoduché *DVRP (RIP, IGRP)* nevytváří vztahy se sousedy
  + problémem jsou routovací smyčky (routing loops)
    - řeší se pomocí
      * definice maximální vzdálenosti (TTL)
      * *split horizon* (neposílá routu na rozhranní přes které přišla)
      * *route poisoning*
      * *hold-down timer* (čekací interval, než je síť stabilní, prodlužuje konvergenci)
* ***Link-State Routing Protocol*** 
  + *OSPF, IS-IS*
  + routery udržují komplexní databázi síťové topologie (vytvořenou pomocí LSA)
  + vyměňují si Link State Advertisements – **LSA**
    - LSA jsou vyvolány nějakou událostí v síti
  + do svého okolí také odesílá Hello pakety, kde zasílá informace o sobě
  + rychle reaguje na změny topologie, ale spotřebovává více pásma (hlavně na počátku zasílá množství LSP) a zdrojů na routeru
  + metrika je komplexní, nejlepší cesta se počítá pomocí ***Dijkstrova algoritmu*** shortest path first - **SPF**
  + pro zlepšení vlastností
    - se rozděluje na menší oblasti
    - hraniční routery posílají sumární cesty
    - využívá multicast
    - číslování LSA
* ***Autonomous System - AS***
  + je skupina IP sítí a routerů, které jsou pod stejnou technickou administrací
  + čísla 1 až 65535
  + privátní rozsah 65512 - 65535
  + AS používá EGP pro komunikaci s jinou AS
  + uvnitř AS se routy naučené z BGP mohou redistribuovat do IGP
* ***Administrative Distance - AD***
  + určuje důvěryhodnost protokolu
  + definuje spolehlivost protokolu a prioritizuje lepší nižším číslem
  + je vlastnost používaná na routrech k určení nejlepší cesty mezi více routovacími protokoly
  + jinak řečeno na routeru může běžet více routovacích protokolů a podle AD se rozhoduje, který se použije
  + na Cisco routrech můžeme měnit defaultní hodnoty
* **konvergence**
  + procesy a čas potřebný pro konverzi směrovacího protokolu
  + konvergence je dosažena ve chvíli, kdy všechny routery mají kompletní aktuální informaci o topologii

## protokol RIP

* *Routing Information Protocol (RFC 1388)*
* metoda, kterou používají směrovače (routery) k tomu, aby zajistily stav síťové konektivity a nalezly nejlepší cestu, přes kterou budou zasílat data
* RIP je protokol typu *router-to-router*, přičemž většina velkých standardů má svůj vlastní, s ostatními nekompatibilní RIP (IP, IPX, DECnet atd.)
* představitel distribuovaného směrovacího protokolu
  + směrovací informace si vyměňují sousedé či malé skupiny směrovačů
* **nejčastější IGP** (*Interior Gateway Protocol*)
* protokol nad UDP
* klasický „vector distance“ = založen na vektoru vzdáleností
  + **ve směrovací tabulce má**:
    - cíl
    - kudy nejlépe k němu
    - vzdálenost
* **metrikou je počet hopů**
  + vzdálenost se měří ve „skocích“ (hop) – přeskok paketu mezi 2 směrovači má délku (typicky) 1
  + **maximální vzdálenost je 15**
    - **16 = nekonečno**
* užívá se pouze pro menší sítě
* **algoritmus**
  + každých 30 s pošle svoji směrovací tabulku sousedovi
  + soused přičte je všem vzdálenostem 1 a porovná se svou tabulkou, změní si záznam pokud:
    - cíl ještě neznal
    - zná k cíli delší cestu
    - cesta k cíli vede přes odesilatele tabulky (zhoršila se, ale nedá se nic dělat)
  + nový směr jen je-li striktně kratší (aby nedocházelo k oscilacím)
* ***hold down***
  + spuštění časovače v případě zavedení nové cesty do tabulky (180s)
  + nedorazí-li v limitu pro tuto cestu alespoň jedna další směrovací informace, cesta ztrácí platnost
* ***split horizon*** 
  + nevrací informace tomu, kdo ji poslal
* ***poison reverse***
  + informace se posílá na všechna rozhraní
* ***triggered update***
  + kdykoliv router změní metriku nějaké cesty, okamžitě vyšle novou směrovací tabulku
* **výhody**
  + jednoduché zavedení
  + velmi rozšířený
  + podpora takřka všemi výrobci
* **nevýhody**
  + pomalá konvergence – každý krok 30 s
  + malá maximální cena – nelze vyjádřit kvalitu linek
  + přenos velkého množství směrovacích informací
  + vznik dočasných cyklů
  + omezená metrika

## Protokol IGRP

* *Interior Gateway Routing Protocol*
* robustnější než RIP – vektor vzdáleností

## Protokol EIGRP

* *Enhanced Interior Gateway Routing Protocol*
* hybridní routovací protocol
* **metrika**
  + **32 bitová**
  + skládá se z
    - vektor vzdáleností
    - zpoždění na lince
    - spolehlivost
    - zatížení
  + většinou se používá pouze ***šířka pásma*** a ***delay***
* používá spolehlivý přenosový protocol RTP (Realiable Transport Protocol)
* proprietární Cisco protokol, rozšíření staršího IGRP
* jedná se o rozšířený (Advanced) ***Distance-Vector Routing Protocol***
* obsahuje i některé vlastnosti z link-state routing protokolu
* **používá DUAL algoritmus** - *Diffusing Update based algorithm*
* oproti IGRP
  + rychlejší konvergence
  + větší rozšiřitelnost
  + smyčky jsou téměř vyloučeny (díky DUAL algoritmu)
  + zasílá pouze změny, neperiodicky
* jedná se o ***classless*** protokol,
  + používá CIDR a VLSM (*variable length subnet masks*)
  + jako masku zasílá délku prefixu pro každý cílový subnet
* sestavuje vztahy se sousedy zvané ***sousedství*** (*adjacency*)
  + sousedy objevuje pomocí ***Hello paketů***, které se zasílají každých
    - 5s na rychlých
    - 60s na pomalých linkách
* aby dva routery mohly být sousedy
  + musí být členy stejného ***AS*** (Autonomous system)
  + mít stejné ***K values*** (konstanty pro výpočet metriky)
  + také stejná verze EIGRP (zatím se nepoužívá)
  + interface na stejném subnetu
  + stejný MD5 hash
* defaultně používá ***autosumarizaci*** přes classful hranice
  + zajišťuje, aby routovací tabulky byly co nejvíce kompaktní
  + můžeme vypnout no auto-summary
    - to je třeba v případě, kdy máme nespojitý adresní prostor
      * při sumarizaci by se zahrnuly i adresy, které jsou mimo naši síť
  + můžeme použít manuální sumarizaci per interface, jako cíl (destination) se nastaví *Null0*
* podporuje IP, IPv6, IPX, AppleTalk
* maximálně zabere (v defaultním nastavení) 50% propustnosti interfacu (údaj získává z bandwidth)
* **základní termíny EIGRP**
  + ***successor***
    - primární routa k cíli, ukládá se do routovací tabulky
    - může být více succesorů ke stejnému cíli
  + ***feasible successor***
    - záložní cesta (backup route)
    - ukládá se do tabulky topologie
    - může být více feasible succesorů ke stejnému cíli, její rd je menší než fd současné nejlepší cesty
  + ***reported distance*** (rd)
    - nebo také *advertised distance* (ad), je nejnižší celková metrika (vzdálenost) podél cesty do cílové sítě (přes succesory) zaslaná sousedem
  + ***feasible distance*** (fd)
    - je rd + cena k dosažení souseda, který zaslal rd
  + ***feasibility condition*** (fc)
    - je dostačující podmínka k dosažení sítě bez smyček, používá se při volbě *succesora* a *feasible sukcesora*
    - říká, že pokud je rd < fd pro určitý cíl, tak leží na cestě bez smyček
* pokud router nemá žádný ***feasible successor***, tak se přepne do aktivního stavu a dotáže se sousedů
* ***stuck In Active*** (SIA)
  + stav, kdy je cesta v aktivním stavu, router odešle ***Query*** a v určitém stavu neobdrží odpověď, v tom případě EIGRP vyřadí tyto sousedy
* routovací tabulky (update směrovací informace) se předávají pomocí ***multicastu*** na 224.0.0.10
  + pokud je to možné
  + jinak se použije *unicast* (opakování nepotvrzeného updatu, rozhranní bez podpory multicastu, ručně zadaní sousedé)
* ***nestejnoměrné vyvažování zátěže*** (unequal cost path load balancing)
  + používá proměnnou variance
  + standardně se zvolí cesta s nejmenší metrikou (FD)
  + pokud je takových cest více (mají stejné FD), tak se použije stejnoměrné rozvažování
  + pokud zadáme variance, tak se použijí všechny cesty, které mají metriku menší než *variance\*minimální metrika do cíle* (metrika <= variance\*FD)
  + použijí se ale pouze cesty, které jsou *feasible successor*, tedy pro které platí RD<FD
    - =cesta od souseda do cíle je menší než nejkratší cesta
* **tabulky pro EIGRP**
  + **routovací** (*routing*)
    - nejlepší routy do destinací
  + **topologie** (*topology*)
    - routovací záznamy do všech destinací
  + **sousedé** (*neighbor*)
    - informace o sousedních routerech (adjacent)
* ***stub Routing*** 
  + používá se pro topologii **Hub and Spoke**
    - model kola - střed kola a paprsky, obdoba topologie hvězda
  + jako **stub** se konfigurují pouze remote routers *(*spoke)
  + jako sousedi *stub routeru* by měli být pouze EIGRP hub routers (distribution)
  + snižuje využití zdrojů, zvyšuje síťovou stabilitu a zjednodušuje konfiguraci stub routeru
  + stub router směruje pakety pouze do sítě, kterou explicitně zveřejnil, proto mu Hub router neposílá Query
  + stub router oznamuje (advertised) pouze summary a connected routes
  + neptáme se stub routeru na jich routy

## Protokol OSPF

* otevřený standard, asi nejrozšířenější IGP protokol ve větších firmách
* jedná se o **classless** protokol / podporuje VLSM
* používá ***Dijkstrův algoritmus*** nejkratší cesty - **SPF**
* díky oblastem podporuje hierarchický síťový design
* používá link-state databázi, která předchází smyčkám
* pro snižování routovacích tabulek se může použít **manuální sumarizace**, ale **automatická není podporována**
* routovací updaty se posílají, pouze pokud jsou potřeba
* využívá ***multicast*** (takže neovlivňuje stanice a ostatní zařízení bez OSPF)
  + 224.0.0.6 - všechny DR a BDR
  + 224.0.0.5 - všechny routery
  + 224.0.0.2 - všechny routery na stejném subnetu
* **nepoužívá TCP**, ale IP protokol 89
  + sám si řeší detekci a opravu chyb
* **podporuje autentizaci**
  + plain text nebo MD5, pro akceptování LSA
* **používá hello protokol**
  + lepší než výměna tabulek v RIP, pomocí něj navazuje sousedství
  + posílá každých 10s
  + aby se routery staly sousedy, *hello paket* musí mít shodné hodnoty
    - číslo oblasti
    - typ oblasti (stub nebo NSSA flag)
    - subnet a subnet mask
    - hello a dead timer
    - autentizační údaje
* LSA se odesílá každých 30min nebo při změně
  + šíří se pomocí flooding (zaplavování)
* **cena linky** pro OSPF cost = 100Mb / bandwidth
* **OSPF router ID** (nastavuje se při restartu) je nejvyšší IP adresa (aktivní) na routeru nebo adresa loopback interfacu (to je doporučeno, protože je stále up)
* na routeru může běžet až 30 OSPF routing proces
* **tabulky pro OSPF**
  + **routovací** (routing)
    - nejlepší routy do destinací
  + **topologie** (topology)
    - routovací záznamy do všech destinací
  + **sousedů** (adjacency)
    - informace o sousedních routerech
  + ***link-state database*** 
    - stejná pro všechny routery, synchronizace pomocí zaplavení LSA, pomocí ní se vytvoří routovací tabulky, obraz síťové topologie ve stromové struktuře
* **DR/BDR routery**
  + ***Designated Router - DR*** 
    - posílá LSA 2 všem sousedům v multi-access
    - je to router (přesněji interface routeru), který se volí v rámci segmentu u multiaccess (speciálními technikami i u NBMA)
    - slouží k redukci síťového provozu
    - DR je zdrojem routovacích updatů
    - udržuje si kompletní tabulku topologie, všechny ostatní routery s ním navazují spojení
  + ***Backup Designated Router – BDR***
    - stává se DR, pokud selže původní DR
    - má druhou nejvyšší prioritu v době volby

## Protokol BGP

* současná BGP verze 4 je definována v RFC 4271
* používá se pro routování mezi AS (mezi IGP jako je OSPF)
* posílají se periodické keepalive (60s) pro ověření konektivity
* každé dva routery, které mají navázané TCP BGP spojení (tedy spolehlivé), se nazývají sousedi (neighbors) nebo peers
* každý router s BGP se nazývá BGP speaker
* v updatech se zasílá *Network Layer Reachability Information* (NLRI)
  + cílový prefix, délka, cesta, next hop
* routovací tabulka může obsahovat více jak 100 000 záznamů, díky tomu je internet decentralizovaný systém
* **podporuje CIDR/VLSM a route aggregation**
* nepoužívá tradiční metriky, ale rozhoduje se podle cesty, politik a pravidel
  + EBGP - External BGP - mezi routery z různých AS
  + IBGP - Internal BGP - v rámci stejného AS

## Protokol IS-IS

* standardizovaný ISO (otevřený standard)
* používá ***Dijkstrův*** ***SPF algoritmus***, oblasti a hierarchickou topologii
* pomocí ***Hello paketů*** sestavuje sousedství
* pro broadcast multi-access používá volbu DR routeru
* jedná se o ***classless protokol*** / VLSM / CIDR
* používá ***sumarizaci***
* podporuje autentizaci
* běží na ***síťové vrstvě OSI*** (layer 3)
* využívá ***Network Service Access Point*** (NSAP)
* **nevyužívá IP protokol**, ale OSI networking
* LSP, Hello PDU, routing PDU se posílají jako OSI pakety (ne IP pakety, do kterých se ale mohou zabalit)
* více časovačů (timers) pro ladění
* méně typů oblastí než OSPF
* více škálovatelný (až 1000 routerů v jedné oblasti)
* menší zatížení CPU, ještě rychlejší konvergence než OSPF
* používá ***Partial Route Calculations*** (PRC) pro výpočet dostupnosti
* hierarchie s Level 1 a 2 routery na rozdíl od oblastí
* používá ***domény*** (obdoba AS u TCP/IP)
  + sekce sítě pod stejnou administrative control
* používá pouze ***point-to-point*** (sériové linky) a ***broadcast network*** (default pro LAN)
* nepodporuje point-to-multipoint a non-broadcast
* ***metrika***
  + cena (cost)
    - max. 1024 pro cestu
    - max 64 pro link
    - default je 10 pro každou linku, nastavuje se ručně
  + pro ne-Cisco také delay (cestovní zpoždění)
  + *expense* (náklady na využití linky)
  + *error* (pravděpodobnost chyby)

## Směrovače

* protokolově závislý
* umožňuje propojení více než dvou částí globální sítě
* spojuje různé sítě pracující však na stejném protokolu a hledá optimální cesty mezi uzly
* přejímá rámce mezi sítěmi a na chvíli vždy spojí ty dvě, mezi kterými probíhá komunikace
* pro každou síť si ukládá tabulku logických adres
* užívá směrovací (routovací) tabulky
  + pomáhají mu v rozhodování
* omezuje výskyt kolizí
* přenosový protokol definován na 3. a 4. vrstvě (např. IP vyžaduje TCP)
* hardwarové zařízení, které přebírá data z lokální počítačové sítě a směruje je na vzdálený cíl po telefonní či jiné dálkové lince (ISDN, optika), popřípadě provádí opačnou funkci
* náročné zařízení umožňující simultánně přebírat a směrovat data z různých zdrojů na různé cíle, obsahuje také řadu bezpečnostních algoritmů zabraňujících probourání vetřelců do lokální sítě
* směrovače také řídí správným způsobem provoz na linkách, korigují chyby atd.

# Propojování a management sítí

Přenosová média, technologie pro různé vrstvy, WIFI, VPN, systémy pro vzdálený přístup, řešení založená na SNMP

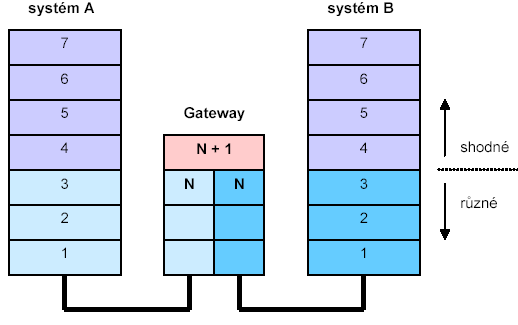
## základní pojmy

* v ideálním případě by měla zcela zaniknout jakákoli specifika jednotlivých sítí, aby se uživatel mohl domnívat, že pracuje v jedné jediné, všezahrnující síti.
* vzájemným propojením dvou či více sítí stejného či různého typu vzniká větší celek, pro který má angličtina velmi výstižné označení: **internetwork**
* problematika vzájemného propojování sítí a jejich součinnosti (v angličtině: **internetworking**) je sama o sobě značně rozsáhlá, rychle se vyvíjí, a je dnes velmi populární
* v současné době existuje řada různých koncepcí toho, jak počítačové sítě navzájem propojovat
* pro jejich pochopení je ale klíčovým momentem poznání toho, že vzájemné propojení je možné realizovat na různých úrovních vrstvového síťového modelu - od fyzické až po aplikační vrstvu
* základní myšlenka vzájemného propojení je naprosto triviální
  + dvě nebo více sítí se propojí prostřednictvím k tomu určených zařízení, obecně označovaných jako **relay**
    - v terminologii iso též: ***intermediary system***

## Přenosová média

* **linková**
  + koaxiální kabel
  + kroucená dvoulinka (nestíněná = UTP, stíněná = STP),
    - kroucení odrušuje elektrický šum ze sousedních dvoulinek apod.
    - ***UTP***
      * nejoblíbenější typ
      * max. délka 1 kabelu je 100 m
    - ***STP***
      * podobná UTP
      * lépe chrání přenášená data (každý vodič je kryt izolací)
* **optické kabely**
  + optická vlákna nesou modulované světelné impulsy
  + vhodné pro přenos velkého objemu dat velmi vysokou rychlostí
  + čistota signálu
* **radiové spoje**
  + elektromagnetické vlny, mikrovlny, infračervené záření
* **družicové spoje**
  + velmi drahé

## Technologie pro různé vrstvy

* pokud se nejvyšší různá vrstva označí jako vrstva n, pak předání dat se může odehrát nejníže ve vrstvě n+1
* brána předávající data ve vrstvě n+1 se označuje jako **brána úrovně n**
* pro některé úrovně existují zavedené názvy:
  + **gateway úrovně 1**: **repeater** (opakovač)
  + **gateway úrovně 2**: **bridge** (most)
  + **gateway úrovně 3**: **router** (směrovač)
    - v protokolu TCP/IP se zde poněkud nešťastně v angličtině obvykle používá slovo gateway
* **opakovače**
  + propojující zařízení přitom může být pouhým zesilovačem elektrických signálů, které jsou v síti přenášeny
  + tak je tomu například v lokálních sítích typu ethernet, které umožňují dosáhnout celkové délky kabeláže až 2,5 kilometru, ale jednotlivé připojovací obvody (tzv. transceivery) jsou schopné generovat elektrické signály s dosahem jen asi 500 metrů
  + pak je nutné sestavovat celé kabelové vedení ze segmentů (souvislých úseků kabelů) délky maximálně 500 metrů
  + jednotlivé segmenty se pak musí spojovat pomocí zařízení, která se v tomto případě nazývají opakovače (**repeaters**), a fungují právě a pouze jako zesilovače elektrických signálů
  + opakovače tedy pracují na úrovni fyzické vrstvy, neboť vzhledem k zesilování elektrických signálů "vnímají" jen jednotlivé přenášené bity, ale nikoli již celé bloky (rámce) dat
* **mosty**
  + nevýhodou opakovačů je skutečnost, že "**propouští**" veškerý provoz z jednoho segmentu do druhého, i když by to vůbec nebylo nutné
  + výhodnější by jistě byl případ, kdyby propojovací zařízení dokázalo správně rozpoznat, která data mohou zůstat "uvnitř" příslušné části sítě (segmentu), a nezatěžovat jimi provoz v ostatních segmentech
  + k tomu je ale nutné, aby propojovací zařízení "vnímalo" celé přenášené rámce a znalo jejich formát natolik, aby dokázalo zjistit fyzickou adresu jejich odesilatele a příjemce
  + to ovšem znamená, že takovéto propojovací zařízení, které se pak nazývá most (**bridge**), musí pracovat na úrovni linkové vrstvy (resp. na úrovni podvrstvy řízení přístupu k médiu (vrstvy mac) v případě lokálních sítí dle standardů ieee 802), kde již lze fyzické adresy příjemce a odesilatele rozpoznat
  + zatímco opakovač nemá paměť a přenášená data resp. signály zpracovává průběžně (je pro ně vlastně "průchozí"), most již pracuje na principu "*store and forward*" (přijmi a předej dál)
  + most tedy z každé strany průběžně přijímá jednotlivé datové rámce, a podle adres v nich se rozhoduje, zda je předá na opačnou stranu či nikoli
  + jednou z nejjednodušších variant je ta, při které most průběžně vyhodnocuje odesilatele jednotlivých rámců, a podle toho, ze kterého směru příslušný rámec přijal, si pak sám odvozuje umístění jednotlivých uzlů
    - záhy se tak sám dokáže "naučit" topologii sítě. v době, kdy ji ještě nezná, jednoduše předává všechny rámce do všech ostatních segmentů.
    - tato metoda je velmi atraktivní proto, že nevyžaduje žádné konfigurování mostu
    - jednotlivé uzly v síti přitom nemusí o jeho existenci vůbec vědět - proto se také takovýto typ mostu označuje jako tzv. **transparentní most** (*transparent bridge*)
    - lze jej ovšem použít jen v takových sítích, které mají přísně stromovitou strukturu, kdy mezi každými dvěma uzly existuje vždy jen jedna jediná cesta
    - pro obecnější topologie jsou pak nutné jiné, složitější algoritmy práce mostů
  + pokud jde o technické provedení, mohou být opakovače i mosty konstruovány tak, aby vzájemně propojovaly jen dva segmenty, nebo také více segmentů najednou
    - pak jde o tzv. **vícevstupové opakovače** (*multiport repeater*) resp. **vícevstupové mosty** (*multiport bridge*).
  + mosty se vyrábí i ve variantě tzv. **vzdálených mostů** (*remote bridge*)
    - od standardní varianty mostů se vzdálené mosty liší v tom, že jde vlastně o dvě relativně samostatné "poloviny" mostu, příznačně nazývané **půlmosty** (*halfbridge*), které jsou mezi sebou vhodně propojeny
      * např. pevným telefonním okruhem, optickým kabelem apod.
    - umožňují propojit dva segmenty sítě, které nejsou fyzicky blízko sebe
    - takto lze například propojit dva segmenty lokální sítě ve dvou objektech na opačných stranách města, přičemž výsledný efekt je takový, že oba segmenty tvoří jedinou "logickou" síť (z pohledu síťové vrstvy a všech vyšších vrstev je totiž existence místních i vzdálených mostů transparentní)
  + některé mosty pak mohou mít i schopnost selektivního filtrování některých rámců v závislosti na jejich odesilateli či příjemci, denní době, intenzitě provozu apod. pak jde o tzv. **routing bridges**, které správcům sítí umožňují regulovat přenosy mezi jednotlivými segmenty
    - umožňují například zakázat v době "špičky" přístup z jednoho segmentu do jiného, a při poklesu intenzity provozu jej pak zase následně povolit
  + **opakovače** jako jednodušší a lacinější (a současně i rychlejší)
    - volí se spíše tam, kde intenzita provozu není velká
  + **použití mostů** (obecně složitějších, dražších a také pomalejších než opakovače) je naopak vhodné tam, kde je potřeba vzájemně propojit segmenty s větší intenzitou provozu tak, aby se navzájem nezatěžovaly více, než je skutečně nezbytně nutné
* **routery**
  + použijeme
    - požadujeme-li, aby si jednotlivé segmenty lokálních sítí zachovaly relativní samostatnost
    - když potřebujeme vzájemně propojit lokální sítě různých typů
    - spojujeme-li dvě lokální sítě přes síť rozlehlou
    - vytváříme-li vzájemné propojení sítí se složitější topologií
  + pracuje na úrovni síťové vrstvy
    - teprve takovéto zařízení totiž "vnímá" vlastní obsah jednotlivých rámců (na úrovni linkové vrstvy)
    - dokáže správně rozpoznat formát jednotlivých paketů, které jsou v rámcích přenášeny
  + **hlavní úkol směrovačů** je shodný s úkolem síťové vrstvy
    - postarat se o doručení paketů od jejich původního odesilatele až ke konečnému příjemci
  + další charakteristickou odlišností směrovačů od mostů je to, že jsou pro ostatní entity na úrovni síťové a linkové vrstvy viditelné
    - mají své adresy, a pakety, které jimi mají projít, jim jsou explicitně adresovány (zatímco mosty zachycují veškerý provoz v každém z připojených segmentů)
    - proto také směrovače zpracovávají méně rámců než mosty, ovšem jejich zpracování je zase o to náročnější
  + pro funkci směrovače je nutné, aby vzájemně propojované sítě používaly stejný protokol na úrovni síťové vrstvy
    - podle něj totiž směrovač rozpoznává odesilatele i adresáta jednotlivých paketů, a rozhoduje o tom, kudy je dále odeslat
    - není ovšem nutné, aby totéž platilo i na úrovni linkové a fyzické vrstvy
      * zde se již konkrétní protokoly a přenosové technologie mohou lišit
      * směrovače jsou dnes obvykle konstruovány tak, aby měly více různých rozhraní (tzv. portů)
* **multiprotokolové směrovače** 
  + požadavek stejného (a tudíž jediného) protokolu v síťové vrstvě je velmi omezující, zvláště v dnešní době, kdy vedle sebe koexistuje celá řada soustav protokolů (kromě iso/osi též tcp/ip, sna, decnet, spx/ipx a další), a uživatelé volají po jejich co nejtěsnější integraci v rámci tzv. heterogenních sítí (tj. sítí, jejichž uzly používají různé soustavy protokolů)
  + **problém heterogenních** sítí lze řešit v principu dvěma způsoby
    - **konverzí protokolů**
    - **směrováním více protokolů současně**
    - řešení prostřednictvím konverzí se ukázalo být značně náročné a nespolehlivé, a proto se prosadila především druhá možnost
  + přední výrobci dnes nabízí tzv. **multiprotokolové směrovače** (*multiprotocol routers*) schopné pracovat současně s více různými protokoly
  + musí být schopen rozpoznat typ paketu, který dostane od linkové vrstvy, a podle toho pak aplikovat ten směrovací algoritmus, který k příslušnému síťovému protokolu přísluší
* **brouter**
  + bridge/router, někdy též: *brouter*
  + jde o zařízení, které se snaží fungovat jako směrovač, a teprve v okamžiku, kdy pro nějaký paket neumí aplikovat směrovací algoritmus, předá původní rámec dál tak, jako by to udělal most
  + výhodou takovéhoto zařízení je pak i to, že se dokáže vyrovnat s takovými protokoly, které vůbec nelze směrovat
* **brána**
  + pokud je potřeba vzájemně propojit sítě zcela odlišných koncepcí, používající zcela jiné soustavy protokolů, je nutné použít propojovací zařízení, schopné provádět nezbytnou konverzi protokolů
  + takovéto zařízení, označované nejčastěji jako **brána** (*gateway*) pak pracuje na takové úrovni, na které je možné příslušnou konverzi zajistit

## Prostředky pro správu sítí

* *NMS*
  + *Network Management Station*
* *Open View*
  + od Hewlett Packard
* *EMS*
  + *Enterprise Management System*
  + podpora většiny výrobců
* ***SNMP***
  + *Simple Network Management Protocol*
* jiné významné nástroje
  + *Unicenter*
  + *IMB Tivoli NetView*
  + *MMC (Microsoft Management Console)*

## 

## SNMP

* slouží k výměně informací, které se člení na jednotlivé proměnné a jež jsou uspořádány do hierarchického stromu, tzv. MIB (*Management Information Base*), podobně jako např. adresářová struktura souborů na disku
* slouží ke komunikaci mezi managery a agenty na řízených uzlech sítě
* možnosti jeho využití jsou ale podstatně větší, a tak stále častěji proniká i do průmyslové automatizace a měřicí techniky
* [asynchronní, transakčně orientovaný protokol založený na modelu klient/server](http://hwgroup.cz/products/charon1/snmp/index.html)
* strana, která posílá požadavky (**snmp klient**), může být např. jednoduchý *snmp browser* či složitý NMS (*Network Management Systém)*
* na straně zařízení je **snmp agent** (snmp server), který na požadavky odpovídá
* výjimku tvoří tzv. **trapy**, které agenti vysílají asynchronně při výskytu jednotlivých události
  + výpadek proudu, větráku, překročení mezních údajů, objevení nového zařízení
* samozřejmě je nutné předem definovat adresu, kam se informace posílá
* pro přenos dat se používá protokol **UDP**
* za standard je odpovědné sdružení IETF (*Internet Engineering Task Force*)
* SNMP je protokolem rodiny TCP/IP
* **klientský program**
  + zvaný síťový manažer
  + vytváří virtuální spojení se SNMP agentem, který běží na sledovaném síťovém zařízení
* **agent**
  + malý program, který monitoruje stav zařízení a poskytuje o něm informace manageru (např. počet zpracovaných paketů/sec atd.)
  + informace, poskytované agentem, jsou uspořádány podle databáze *MIB (Management Information Base),* která svojí strukturou odpovídá danému zařízení
  + veškerá komunikace mezi manažerem a agentem je vykonávána prostřednictvím *Network Management Protocol*
  + v prostředcích
    - routery
    - přepínače
    - rozbočovače
    - servery (W2000, Unix, Linux, …)
    - pracovní stanice
    - tiskárny
    - UPS
* **získání informací o daném zařízení**
  + manažer vyšle požadavek na dané zařízení (pooling aktivita) a vyhodnotí informace poskytované agentem
    - automaticky (v určených intervalech)
    - vyžádání správcem
  + agent detekuje stavy definované hodnoty a vyšle trap – možnost provedení definovaných akcí
    - HW porucha
    - zátěž zařízení nad stanovený limit
    - pokus o neznámý přístup
    - ukončení automatické aktualizace
  + kontinuální monitorování je z důvodu zátěže sítě nereálné
  + efektivita je zajištěna kombinací *pooling* a trap aktivity managera a agenta
* SNMP pracuje na modelu žádost-odpověď
  + monitorovací dotazy
  + prahové dotazování
  + výkonové dotazování
* agent ukládá informace o objektech do datových záznamů **MIB** (*Management Information Base*)
  + objekty Cisco MIB
    - konfigurace
    - frekvence chyb rozhraní
    - šířka pásma (echo)
    - hustota síťového provozu
    - nedosažitelné adresy
    - data SNMP (vlastní režie)
* konzole SNMP získává velké množství dat
* po nastavené době z nich vypočte různé agregace a data MIB vymaže (přepíše je novými)
* příkazy protokolu SNMP (6 základních příkazů)
  + *Get* - načtení od agenta
  + *GetNext* - načtení dalších instancí
  + *GetBulk* - čte první a všechny následující instance
  + *Set* - nastavení hodnot v agentu (např. prahové hodnoty)
  + *Trap* - agent oznamuje vznik události
  + *Inform* - nařízení předávat data i jinam
* **výhody**
  + správa Internetové sítě jediným nástrojem
* **nevýhody**
  + spotřeba systémových prostředků
  + rozdílná podpora standardů výrobci
  + práce týmu správců
  + vyšší koncová cena

## 

## RMON (Remote Monitoring)

* doplňuje SNMP
* odlišnosti
  + je postaven na zvláštních strojích
  + nečeká na dotazování dat
  + umožňuje podrobnější sběr údajů
* je dražší
  + použití jen na kriticky důležitých linkách

## 

## CiscoWorks 2000

* balík pro správu sítě
* asi 20 aplikací postavených na infrastruktuře SNMP a RMON

# Principy operačních systémů

Základní rozdělení, procesy, správa procesů a systémových zdrojů, metody přidělování CPU, metody přidělování paměti, synchronizace procesů

## Rozdělení

* pro pracovní stanice
  + síťové
* serverové

## Základní pojmy

* **úloha**
  + souhrn činností, kterou je třeba provést k uspokojení požadavků uživatele
* **proces**
  + výpočet (každá úloha se rozpadá na několik procesů)
  + obvykle identifikován pomocí PID
* **procesor**
  + technické zařízení, které vykonává posloupnost instrukcí
* **základním úkolem správy procesoru** je jeho přidělování procesům tak, aby při provozu vznikaly minimální ztráty
* správa procesů má 3 funkce
  + **plánovač úloh**
    - z přijatých úloh vytváří procesy a rozhoduje, kdy bude která úloha zpracovávána
  + **plánovač procesů**
    - rozhoduje, kterému z připravených procesů bude procesor přidělen a jak dlouho jej bude moci používat
  + **synchronizace procesů**
    - např. tiskárnu používá 1 proces, a teprve po něm může tiskárnu použít 2. proces
* obecně může být každý proces v 1 z 3 stavů
  + **probíhající**
    - procesu je přidělen procesor (provádí se posloupnost instrukcí)
  + **čekající** 
    - proces čeká na událost, která má nastat před dalším pokračováním jeho činnosti
  + **připraven**
    - proces je připraven ke zpracování (čeká jen na přidělení procesoru)

## METODY PŘIDĚLOVÁNÍ CPU

* v jednoduchém OS pokud jediná spuštěná úloha musí čekat, je blokováno CPU
* v multiprogramovém OS je možno tento čas využít produktivně
  + v paměti je současně uchováváno více úloh
  + v okamžiku, kdy musí běžící úloha čekat, je CPU přiděleno jiné
* běh procesu je složen z cyklu spuštění na CPU a čekání na I/O, probíhající proces přebíhá mezi těmito dvěma cykly
* při posledním CPU cyklu dojde k systémovému volání, které vede k ukončení procesu
* V okamžiku, kdy by mělo CPU být spuštěným procesem blokováno, OS ho musí přidělit jinému procesu z fronty připravených – tento výběr provádí plánovač CPU
* každý proces je ve frontě reprezentován zejména svým PCB
  + fronta připravených může být implementována jako:
    - FIFO
    - fronta s prioritou
    - strom
    - neuspořádaný seznam
* **Dispečer**
  + Část OS, která zajišťuje přidělování CPU
  + funkce dispečera jsou:
    - přepínání kontextu
    - přepínání mezi uživatelskými mody
    - nalezeni místa, kde byl uživatelský program přerušen a znovu ho spustit
* **Preemptivní plánování**
  + K předělení CPU může dojít kvůli některému z následujících důvodů:
    - jestliže proces přechází ze stavu probíhající do stavu čekající, např. kvůli I/O požadavku nebo kvůli čekání na dokončení synovského procesu
    - jestliže proces přechází ze stavu probíhající do stavu připraven, např. kvůli přerušení
    - jestliže proces přechází ze stavu čekající do stavu připraven, např. kvůli dokončení I/O operace
    - jestliže je proces ukončen
* **Nepreemptivní plánování**
  + Je-li v případě nepreemptivního plánování CPU přidělen procesu, ten ho opouští, pouze pokud je ukončen nebo čeká – např. na I/O operaci
  + tato metoda plánování je užita v prostředí Microsoft Windows
  + nepotřebuje totiž speciální hardware jako preemptivní plánování (časovač apod.)
* **Kritéria přidělování CPU**
  + Různé přidělovací algoritmy mají různé vlastnosti a mohou upřednostňovat třídu procesu před jinými
  + Kritéria užitá pro posouzení plánování CPU jsou:
    - využití procesoru
    - propustnost
    - doba obrátky
    - doba čekání
    - doba odezvy
  + Každý OS se snaží
    - maximalizovat využití CPU a propustnost
    - minimalizovat dobu obrátky, čekání a odezvy
  + Často výhodnější minimalizovat průměr, než minimum či maximum
    - v interaktivních systémech je výhodnější minimalizovat rozptyl doby odezvy než její průměr
* Příklady algoritmů
  + **FCFS** (First Come, First Served Scheduling)
    - **FIFO –** první přišel, první dostal – nejjednodušší algoritmus, nepreemptivní
  + **SJF** (Shortest Job First Scheduling)
    - První jde vždy úloha s nejkratším CPU cyklem, v případě dvou stejných rozhodnuto pomocí FCFS
    - Může být preemptivní i nepreemptivní – nová nejkratší úloha se buď zařadí do fronty, nebo přeruší stávající delší
    - SJF využívá **plánování podle priority**, ta je buď interní – čas CPU, nebo externí – nastavená OS nebo uživatelem podle důležitosti procesu
    - Procesy ve frontě „stárnou“ a získávají vyšší prioritu, aby nemohlo dojít k jejich neobsloužení
  + **RR** (Round Robin Scheduling)
    - Plánování cyklickou obsluhou
    - Fronta připravených implementována FIFO frontou procesu a nový proces je vždy zařazen na její konec – podobné FCFS
    - Procesy jsou ve vykonávání přerušovány podle definované jednotky časového kvanta – zabírají CPU právě jen po dobu jeho trvání, pak dojde k vystřídání s jiným
    - Možnost preemptivního plánování – přerušení běžícího procesu s cyklem větším, než časové kvantum, pokud přijde proces kratší
  + **MQS** (Multilevel Queue Scheduling)
    - Procesy rozděleny do skupin, např. na interaktivní a na pozadí
    - Každá skupina má svou vlastní frontu s vlastním plánovacím algoritmem
    - Plánování mezi frontami obvykle jako preemptivní plánování s pevnou prioritou (interaktivní procesy mají vyšší), popř. podle daných časových intervalů pro každou frontu
  + **MFQS** (Multilevel Feedback Queue Scheduling)
    - Umožňuje procesům přesun mezi frontami, velice flexibilní, významný algoritmus

## správa paměti

* úkolem modulu správy paměti (operační) je monitorovat stav každého paměťového místa, sledovat, zda je volné, nebo přidělené nějaké úloze
* tato část OS určuje strategii přidělování paměti a zvolenou strategii technicky realizuje
* vhodná volba této strategie velmi výrazným způsobem ovlivňuje propustnost celého systému
* **přidělování jediné souvislé oblasti paměti**
  + tato jednoduchá metoda používaná zpravidla u monoprogramových systémů nevyžaduje použití žádných zvláštních technických prostředků.
  + celá operační paměť je rozdělena na tři souvislé oblasti
  + operační systém je trvale uložen na začátku paměti
  + úloze je přiřazen celý zbytek paměti, přestože ve skutečnosti bude využívat jen její část
  + část paměti, kterou úloha pro svou práci nepoužije, zůstává nevyužita
  + v případě této strategie přidělování paměti je modul správy paměti velmi jednoduchý
    - operační paměť je vždy přidělena jediné úloze a po jejím ukončení je opět celá uvolněna
  + **nevýhody**
    - vždy může být zpracovávána jen jedna úloha, tzn., že není plně využit čas CPU ani dalších zařízení
    - jestliže spuštěná úloha má malé paměťové nároky, zůstává většina paměti nevyužita
    - nemůže být provedena úloha, jejíž paměťové nároky jsou větší než kapacita paměti
    - jestliže počítačový systém zpracovává v dané čase jedinou úlohu, je velmi neefektivní
* **přidělování paměti po sekcích**
  + metoda je založena na myšlence rozdělení operační paměti na samostatné paměťové oblasti (sekce), přičemž každá oblast obsahuje paměťový prostor jedné úlohy
  + v případě této strategie musí OS
    - sledovat stav každé sekce
    - rozhodovat, které úloze bude paměť přidělena
    - provádět vlastní přidělení (z volných sekcí vybrat sekci nejvhodnější velikosti)
    - uvolnit paměť po ukončení úlohy pro další přidělení
  + protože je v tomto případě paměť obsazena více úlohami, je nutné, aby systém obsahoval mimo jiné i mechanismus ochrany proti narušení paměťového prostoru jedné úlohy jinou
* **nevýhody**
  + fragmentace paměti
* **řešení problému fragmentace**
  + dynamické přemísťování sekcí
    - slučování všech volných oblastí paměti do jedné souvislé (zhušťování = defragmentace)
  + stránkování paměti
    - adresový prostor úlohy je rozdělen na stejné úseky (stránky)
    - paměť je rozdělena na bloky (stejné délky jako stránky)
    - stránky zůstávají vzhledem k uživatelskému programu souvislé, ale odpovídající bloky spolu fyzicky souviset nemusí
  + segmentace paměti
    - je logické seskupení informací, které se berou v úvahu jako entita

## stránkování paměti

* důležitým problémem je velikost stránky
* v případě, kdyby byl rozsah stránky příliš velký, nebyl by zřetelný rozdíl mezi strategií stránkování a dělením paměti na sekce
* při malém rozsahu stránky by bylo stránek hodně a tím i položek tabulek příliš mnoho
* je třeba zabezpečit mechanismus, který umožňuje technicky realizovat promítání paměťového prostoru úlohy do fyzické paměti
  + k tomu účelu musí systém obsahovat různé tabulky
  + tabulky mohou být realizovány buď v části operační paměti, nebo jako funkce speciálního technického vybavení
* pro adresový prostor každé úlohy existuje jedna tabulka, která obsahuje pro každou stránku jeden záznam tvaru „*stránka / přiřazený blok“*
* v systému dále existuje tabulka obsazení bloků paměti, která obsahuje pro každý paměťový blok informaci, zda je volný či přidělen
* rozhodování o přidělení paměti konkrétním úlohám provádí správce paměti
* úloze je přidělena množina volných bloků paměti
* určí se počet stránek potřebných pro úlohu, a jestliže je v operační paměti odpovídající počet volných bloků, je rozhodnuto o jejich přidělení
* vlastní přidělení se potom realizuje tak, že všechny stránky úlohy se zavedou do přidělených bloků paměti a provedou se příslušné zápisy do tabulky stránek a tabulky bloků paměti
* v okamžiku ukončení úlohy se jí přidělené bloky označí v tabulce bloků jako volné
* **výhody**
  + dochází k odstraňování fragmentace
  + díky stránkování se zvyšuje počet současně spustitelných úloh
* **nevýhody**
  + tabulky zaujímají část paměti
  + paměť obsahuje údaje, které se právě nevyužívají (stejně jako u předchozích metod)

## Stránkování na žádost (virtuální paměť)

* tato metoda dává k dispozici téměř neomezené množství paměti
* virtuální paměť je ve svém principu větší než operační paměť
* je-li naplánováno zpracování úlohy, je ve skutečnosti do paměti zavedena pouze první (startovací) stránka
  + ostatní stránky se do paměti zavádí až v okamžiku, kdy je program požaduje
  + protože každá stránka je do paměti zavedena až v okamžiku výpadku stránky (program požaduje použití zatím nezavedené stránky), je zaručeno, že nepotřebné údaje nebudou do paměti zavedeny
* problém nastává, jestliže dojde k zaplnění paměti. V tomto okamžiku je možné další stránku zavést až po uvolnění některého bloku paměti
* důležitým úkolem OS je při použití této strategie ošetření výpadku stránky (*page fault*)
  + jestliže úloha požaduje zavedení další stránky, je třeba zjistit, zda je tato stránka již zavedena v paměti
  + jestliže ano, lze ji samozřejmě použít
  + jestliže ne, je třeba zjistit, zda je v paměti nějaký volný blok
  + v případě jeho nalezení je požadovaná stránka zavedena a může pokračovat další zpracování úlohy
  + v opačném případě je třeba vybrat stránku, která bude odstraněna, uložit ji na disk a na její místo zavést nově požadovanou
  + nejvíce problémů může nastat při odstraňování stránky
    - systém se musí pokusit eliminovat situaci, při které dojde k zahlcení (právě uvolněná stránka je bezprostředně po opuštění paměti znovu zaváděna – v nepříznivém případě může tato situace vést až ke zhroucení systému)
  + proto je kladen velký důraz na výběr odstraňované stránky
    - pro každou stránku v paměti musí existovat příznak, který určuje, zda byla či nebyla v daném časovém intervalu použita
* **výhody**
  + všechny výhody strategie stránkování
  + umožňuje efektivnějším způsobem využívat operační paměť
  + umožňuje zpracovávat větší počet úloh než předchozí metody
* **nevýhody**
  + složitější technické prostředky
  + nutnost dbát na to, aby problémy spojené se stránkováním nezabraly většinu času procesoru
* nejprve se zadanou úlohou zabývá modul spooling, který zařazuje úlohy tak, aby mohly být zpracovány plánovačem úloh (uložení úlohy ve vnitřní formě na disk)
* úkolem plánovače úloh je sledovat všechny úlohy v systému a vybrat tu, která bude připravena ke zpracování
  + z ní vytváří odpovídající procesy a přiděluje jim potřebné prostředky
  + plánovač procesů podle určité strategie přiděluje jednotlivým procesům procesor
* posledním prvkem modulu přidělování procesoru je dispečer sledující stav každého procesu a provádějící potřebné změny stavu
* mimo jiné také realizuje synchronizaci procesů a jejich vzájemnou komunikaci
* důležitá je vhodná volba strategie výběru plánovačem úloh
* úkolem této strategie je vybírat jednotlivé úlohy ze stavu přijat a převádět je do stavu připraven
  + rozčleňovat je na procesy a přidělovat jim požadované prostředky
* jednoduchá situace je v monoprogramovém operačním systému
  + jakmile je procesor přidělen některému procesu (úloze), je uvolněn až po jeho ukončení
  + i zde je ale třeba určitým způsobem plánovat práci systému
* **strategie plánování úloh**
  + **FIFO**
    - nejjednodušší, úlohy jsou zpracovávány v pořadí, v jakém byly systému zadávány
  + **SJF**
    - *shortest job first*
    - k provedení vždy vybírá úlohu, u které je uveden nejkratší odhad doby provádění
  + plánování úloh se znalostí budoucí situace
    - např. je-li známo, že za krátkou dobu budou systému předány ke zpracování dvě krátké úlohy, je výhodnější nezahajovat práci hned, ale počkat až budou tyto úlohy k dispozici
    - tato metoda má nejlepší výsledky ale i mnoho problémů
    - málokdy lze dopředu určit další vývoj a uvedené doby průchodu úloh jsou vždy pouhým odhadem skutečné situace, problémem je také zpracování časově náročných úloh, protože ty mohou na zahájení práce čekat velmi dlouho, atd.
* v multiprogramování je každé úloze přidělen malý časový interval (časové kvantum) a v případě, že je v operační paměti zavedeno více úloh, jsou jim časová kvanta přidělována postupně
  + tímto způsobem lze efektivně využívat času, kdy některá úloha čeká na dokončení I/O operace
* jakmile plánovač úloh připraví určitou skupinu úloh, které budou prováděny, začne pracovat plánovač procesů, který pracuje na nižší úrovni
* tento prvek plní několik funkcí
  + sleduje stav jednotlivých procesů
  + rozhoduje, kterému procesu bude přidělen procesor
  + v kterém okamžiku a na jak dlouho,
  + provádí skutečné přidělování procesoru vybranému procesu a také realizuje uvolnění procesoru
* nejdůležitějším úkolem plánovače procesů je výběr procesu, kterému bude přidělen procesor a určení doby, po jakou bude tento proces moci procesor využívat
* délka zpracování procesoru je ovlivněna řadou faktorů
  + např. proces byl v průběhu zpracování ukončen nebo požaduje komunikaci s některým periferním zařízením,
  + v průběhu zpracování mohlo dojít k chybě nebo procesor vyčerpal přidělené časové kvantum
* ve všech těchto případech je proces nucen opustit procesor a podle příčiny je přiveden buď do stavu čekající, nebo do stavu připraven
* ve stavu připraven se tak vytváří fronta procesů, které čekají na další přidělení procesoru

## Správa periferních zařízení

* **úkolem** je
  + sledovat stav všech připojených zařízení
  + rozhodovat o tom, které periferní zařízení bude přiděleno kterému procesu a na jak dlouho
* **druhy periferních zařízení**
  + I/O zařízení (vstupní/výstupní)
    - klávesnice, scanery, tiskárny, plotery, …
  + paměti
    - chovávají informace

## Správa souborů

* **správa souborů** řídí a realizuje ukládání informací (v podobě souborů) a jejich zpětné vybavení

# Souborové systémy a logická struktura dat

Principy, atributy souborů, operace se soubory, mapování souboru do virtuální paměti, vnitřní struktura souboru, sekvenční a přímý přístup k souboru, VFS v linuxu, vynucené a nevynucené zamykání souboru

## soubor

* základní logická organizační jednotka na disku
* organizovaná množina dat s určitými vlastnostmi
* vystupující pod samostatným, jednoznačným názvem
* může obsahovat data v textové či binární formě
* **atributy**
  + jméno – symbolické jméno souboru v lidsky čitelné podobě
  + typ – pro systémy, které podporují různé typy souborů
  + lokace – ukazatel na zařízení a umístění souboru na tomto zařízení
  + velikost – součásti tohoto atributu je aktuální velikost souboru (v bytech, slovech nebo blocích) a jeho maximální možná velikost;
  + Soubor má v různých systémech různé atributy, typicky:
  + Ochrana – informace o ochraně přístupu k souboru definují, kdo ho může číst, kdo spouštět, kdo zapisovat atd. (pouze pro čtení, skrytý soubor, systémový soubor, soubor k archivování)
  + Datum, čas a uživatelská identifikace – tyto informace mohou být uloženy po vytvoření, poslední modifikaci, posledním užití souboru
* pro lepší organizaci dat existuje struktura zvaná **adresář (~ složka)**
  + organizační jednotka v souborovém systému na disku počítače
  + může obsahovat další adresáře a soubory.
  + organizované do stromové struktury
* adresář nejvyšší úrovně se označuje jako **kořenový adresář** (root directory)
* díky hierarchickému uspořádání adresářů lze jednoznačně určit každý soubor
* k tomu slouží
  + **absolutní cesta souboru** 
    - řetězec se jménem se vytvoří pospojováním všech adresářů, počínaje kořenovým adresář, přes které je nutno projít k danému souboru
    - zakončeno jménem souboru
    - jména adresářů jsou oddělena značkou zpětného či dopředného lomítka
  + **relativní cesta souboru**
    - závisí na aktuálním adresáři
    - popisuje cestu k souboru nikoli z kořenového adresáře, ale z adresáře aktuálního
* **virtuální paměť** každého procesu, který sdílí soubor, obsahuje odkazy na stejné stránky fyzické paměti
  + na stránky, ve kterých je uložena kopie sdíleného souboru
  + přístup procesu ke sdílenému souboru je koordinován systémem, aby byl zajištěn vzájemně jedinečný přístup k souboru

## operace se soubory

* **vytvoření**
  + nejdříve je třeba najít pro soubor dostatečně velký prostor na odkládacím zařízení
  + musí být vytvořen záznam pro nový soubor v adresáři – do tohoto záznamu, kde je uloženo jméno souboru a jeho uložení v systému souborů
* **zápis**
  + k zápisu souboru je prováděno systémové volání, které vyžaduje specifikaci jména souboru a informace, která má být do souboru uložena
  + po přijetí jména souboru systém vyhledá adresář a v něm informaci o umístění souboru
  + systém musí uložit ukazatel zápisu na místo v souboru, kde má být zápis proveden
* **čtení**
  + systémové volání vyžaduje specifikaci jména souboru a informaci o tom, kde v paměti je uložen následující blok souboru
  + je vyhledán patřičný adresář a v něm patřičný záznam, podle něj je nastaven ukazatel čtení na místo, od kterého má být provedeno následující čtení – kdykoli má nastat čtení souboru, je aktualizována pozice ukazatele čtení
  + protože většinou je soubor buď čten, nebo zapisován, mnoho systémů užívá pouze jeden ukazatel aktuální pozice, čímž je šetřen prostor a redukována složitost systému
* **přemístění v souboru**
  + je vyhledán adresář a v něm patřičný záznam
  + ukazatel aktuální pozice je nastaven na patřičnou hodnotu
  + přemístění pozice příští vstupní nebo výstupní operace v souboru nepotřebuje volat žádnou I/O operaci
  + tato akce je známa jako **seek**
* **smazání**
  + je nutné vyhledat adresář, ve kterém soubor leží
  + uvolnit celý úložný prostor, který byl souboru přiřazen
  + smaže se položka záznam o daném souboru v adresáři
* **vypuštění souboru**
  + uživatel chce ponechat nastavené všechny atributy souboru a pouze vyprázdnit jeho obsah
  + tato funkce umožňuje zachovat všechny atributy souboru (s výjimkou velikosti), která je nastavena na nulu
* Těchto šest základních operací představuje minimální sadu nutných instrukcí
  + Příklady dalších potřebných příkazů:
    - připojení nových dat na konec souboru (append)
    - přejmenování souboru (rename)
    - operace pro zjištění velikosti souboru
    - operace dovolující nastavit vlastníka souboru

## Vnitřní struktura souboru

* Některé OS využívají množinu speciálních operací, které manipulují se soubory vybraných datových typů
  + tato myšlenka má nevýhodu v těžkopádnosti a přílišné velikosti kódu OS, pro každou podporovanou strukturu souboru musí OS obsahovat kód pro práci s ní
* Najít interně nějaký offset uvnitř souboru může být pro OS komplikované
  + všechny diskové I/O operace pracují v jednotkách jednoho bloku (fyzický záznam)
  + všechny bloky jsou stejné velikosti
  + je nepravděpodobné, že velikost fyzického záznamu bude přesně stejná jako velikost požadovaného logického záznamu – logické záznamy jsou proměnné délky
  + Packing uložení logických záznamů do fyzických je řešením tohoto problému
* Délka logického záznamu, délka fyzického záznamu a technika packingu
  + Určují, kolik logických záznamů se vejde do jednoho fyzického bloku
    - uložení může provést buď uživatelský program, nebo operační systém
  + nebo může být soubor definován jako sled bloků
    - všechny základní I/O operace pracují na těchto blocích
    - konverze logického záznamu do fyzického bloku je potom softwarový problém

## METODY PŘÍSTUPU k souboru

* **Sekvenční přístup**
  + informace v souboru jsou zpracovávány jeden záznam za druhým
  + velice častý přístup
  + operace prováděné na souboru jsou čtení a zápis
  + operace čtení čte následující část souboru a automaticky posune ukazatel následující I/O operace
  + stejně append, operace zápis provede zápis informací na konec souboru a automaticky posune ukazatel následující I/O operace na nový konec souboru
* **Přímý přístup**
  + soubor sestává z logických záznamů pevné délky
    - umožňuje tak programu číst nebo zapisovat záznamy rychle, ne v částečném pořadí
    - je založena na diskovém modelu souboru, protože disky umožňují náhodný přístup k libovolnému bloku souboru
    - při přímém přístupu je soubor představován očíslovanou sekvencí bloku nebo záznamu
  + soubory s přímým přístupem umožňují rychlý přístup k velkému množství informací
  + operace se soubory musí být modifikovány tak, aby obsahovaly číslo bloku jako svůj parametr

## Zamykání souborů

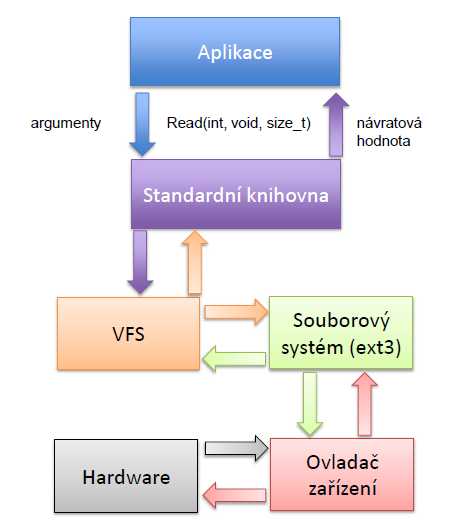
* Zamknutím souboru zabráníme přístupu jiných procesů do otevřeného souboru
* **Nevynucené zamykání**
  + Vyžaduje spolupráci všech procesů přistupujících k souboru.
  + Proces se pokusí získat zámek, pokud se mu to provede, drží jej až do uvolnění.
  + Pokud se mu to nepovede, bude čekat na uvolnění nebo skončí chybou.
  + Operace: čtení, zápis atd. nejsou zámkem ovlivněny
* **Vynucené zamykání**
  + Nevyžaduje spolupráci procesů, ovlivňuje přímo souborové operace.
  + Ty se buď blokují do odemčení, nebo hned selžou

## souborový systém

* fyzické uspořádání na disku zajišťuje takzvaný **souborový systém** (filesystem)
  + udržuje informace o umístění souborů na disku
    - souborový systém je realizován softwarem, který je součástí operačního systému
    - operační systém také uživateli zprostředkuje rozhraní pro základní práce se soubory
  + udržuje vazby souborů na jejich nadřízené adresáře
  + udržuje další metadata o souborech
    - délku
    - datum a čas změny
    - datum a čas vytvoření
    - vlastníka
    - přístupová oprávnění
  + poskytuje služby pro práci se soubory a adresáři
    - vytváření
    - přesouvání
    - nulování
    - připojování
    - úpravu
* rozlišujeme souborové systémy
  + **diskové** 
    - navrženy pro správu souborů na datových zařízeních, nejčastěji diskových jednotkách
    - *FAT12, FAT16, FAT32, NTFS, ext2, ext3, ZFS, ISO 9660…*
    - některé systémy provádějí žurnálová ní
  + **flashové**
    - navrženy pro správu souborů na flashových pamětích
    - tradiční diskové systémy nejsou vhodné protože
      * omezují náhodný přístup (flashovým nevadí)
      * nesnižují opotřebení (flashové systémy rozprostírají data, aby snížili opotřebení)
  + **databázové**
    - u souborů uchovávají i množství dalších metadat
      * téma, autora…
  + **síťové**
    - poskytují klientům přístup k souborům na serveru
    - *SMB*
  + **speciální**
    - dynamické systémy vytvářené softwarem
    - pro účely komunikace mezi aplikacemi, systémem
    - často v Unixu
    - *procfs*
      * poskytuje informace procesech OS

## VFS – Virtual file system

* VFS je abstraktní rozhraní v Linuxu
* Všechny soubory jsou ve VFS
* VFS má podobu zobecněného stromu
* VFS má jeden kořenový systém
* Do VFS lze připojovat různé konkrétní systémy
* Lze vytvářet pevné odkazy
* Lze vytvářet symbolické odkazy
* Lze vytvářet speciální soubory
* Pravidla pro tvorbu názvů jsou velice volná



příklady

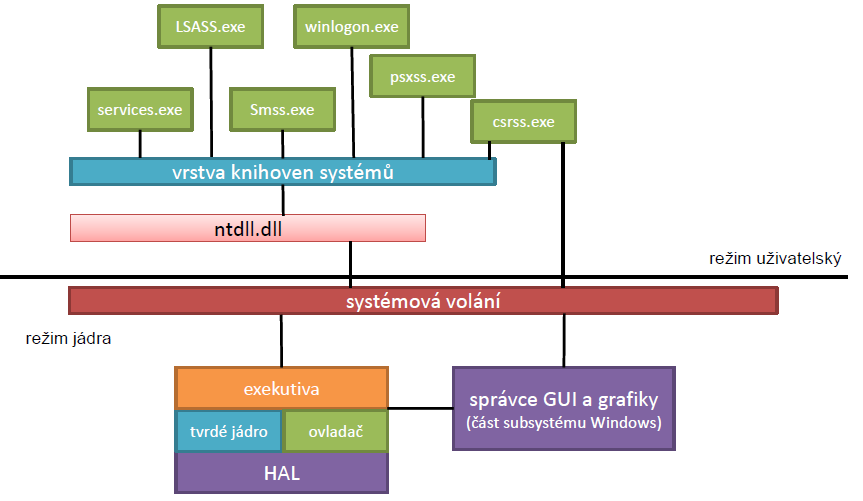
* ***FAT***
  + používá se v PC a na většině paměťových karet, disketách…
  + relativně jednoduchý
  + nízký výkon
  + široká podpora napříč OS
  + v novějších verzích (v Win95) podporuje dlouhá jména souborů
  + nepodporuje oprávnění ani symbolické odkazy
  + řešen spojovým seznamem
  + maximální
    - velikost souboru 4GB
    - délka názvu souboru 255 UTF-16 znaků
* ***NTFS***
  + výchozí souborový systém od Windows NT
  + mnohokrát lepší než FAT
    - vyšší výkon
    - vyšší spolehlivost
    - lepší využít diskového prostoru
    - podpora metadat
    - podpora oprávnění
    - podpora žurnálování
  + podporuje
    - hard-linky pro soubory
    - symbolické linky pro soubory i adresáře
  + od Windows Vista podporuje transakce
  + maximální
    - velikost souboru 16EB
    - délka názvu souboru 255 znaků
  + podporuje kompresy
* ***ext2***
  + pro Linux
  + portován pro jiné systémy (v různé kvalitě a rozsahu)
  + podporuje hard-linky a symbolické linky
  + maximální
    - velikost souboru 2TB
    - délka názvu souboru 255 znaků
* ***ext3***
  + rozšíření *ext2* o
    - žurnálování
    - vyšší spolehlivost
  + zpětně kompatibilní s ***ext2***
* ***xfs***
  + vysoce výkonný žurnálovací systém
  + přímá podpora pro RAID prokládání
  + podporuje zpožděnou alokaci
    - zvyšuje výkon
    - snižuje fragmentaci
  + podporuje vlastní atributy
  + maximální
    - velikost souboru 8EB
    - délka názvu souboru 255 znaků
* ***reiserfs***
  + žurnálovací systém ze světa Linuxu
  + podporuje hard-linky a symbolické linky
  + maximální
    - velikost souboru 8TB
    - délka názvu souboru 255 znaků
* ***ISO 9660***
  + používaný u CD a DVD – jen jednou zapsat pak už jen číst
  + pracuje s rámci o velikosti 24B a sektory (98 rámců = 2352), sektor obsahuje 2048B
  + běžných dat, zbytek pro režijní data (synchronizace atd.)
  + maximálně
    - 8 úrovní adresářů
    - adresářů 65 535
    - velikost souboru 2GB
* ***NFS***
  + nejpoužívanější síťový FS pro Linux
  + implementace složena z
    - serverové
    - klientské části
  + komunikace pomocí volání vzdálených procedur
  + původně podporován jen UDP nyní i TCP
  + celkem 18 procedur (operací), všechny synchronní chování
  + cesty k souborům se rozkládají na jednotlivé komponenty
    - (adresářové úrovně)
  + maximální
    - velikost souboru 16EB
    - délka názvu souboru 255 znaků
  + Max. délka cesty 1024B
  + Podporuje všechny typy souborů

# OPERAČNÍ SYSTÉMY Windows

Charakteristika mikrojádra a monolitického jádra, architektura Windows a jeho subsystémy, systémové procesy - smss.exe, csrss.exe, winlogon.exe, svchostl.exe atd.

## Architektura jádra

* **Mikrojádro**
  + implementuje jen vybrané základní mechanismy
    - virtuální paměť
    - plánování vláken
    - obsluha výjimek
    - zasílání zpráv mezi procesy
  + ostatní komponenty běží v uživatelském režimu
    - zajištěna větší stabilita
    - méně kritického kódu
  + všechny pokročilé funkce v uživatelském režimu
    - časté přepínání mezi režimem jádra a uživatelským - může vést ke zpomalení systémů
* **Monolitické jádro**
  + velké jádro obsahuje většinu komponent
    - souborové systémy
    - správa procesů
    - síťová komunikace
    - bezpečnostní model
  + sdílejí jeden virtuální adresový prostor
    - zrychlení vzájemné komunikace
    - možnost nechtěného ovlivnění a poškození datové struktury
    - teoreticky nižší stabilita a bezpečnost
    - teoreticky vyšší výkon
* Jádro Windows NT řadíme spíše k monolitickému jádru
* **Hardware Abstraction Layer**
  + nejnižší úroveň jádra
  + úkol:
    - odstínění součástí OS a aplikací od specifik hardware
    - poskytovat rutiny pro komunikaci periferií a vyšších vrstev
  + kód HAL uložen v hal.dll
* **Tvrdé jádro**
  + implementace mechanismů pro vyšší vrstvy
  + algoritmus plánování vláken na procesoru
  + odložené volání procedur (DPC)
  + základní synchronizační metody
  + práce s hardwarovým přerušením
  + část obsluhy systémových volání
* **Ovladač**
  + umožňuje vrstvě exekutiva komunikovat s různými typy hardware
  + spustitelné soubory \*.sys
* **Executiva**
  + I/O zařízení a další komponenty
  + využívá tvrdé jádro
  + realizuje složitější mechanismy, které jsou přes systémová volání nepřímo využívána obyčejnými aplikacemi v uživatelském režimu
  + implementována v hlavním modulu jádra
  + ntoskrln.exe nebo ntkrnlpa.exe



## Microsoft Windows

* jedná se o diametrálně nový typ operačního prostředí lépe přizpůsobeného komunikaci člověk-počítač
* všechny verze tohoto prostředí až po Windows95 lze považovat za pouhé nástavby systému MS DOS, který však značně rozšířili a postupně pak z něj „udělali skutečný operační systém"
* **mezi hlavní přínosy patřilo**
  + multitasking
    - současné spouštění a vykonávaní více programů
  + prostředky mezi úlohové komunikace
  + dynamicky linkované knihovny
  + správa pamětí
  + sdílené technické prostředky
    - display, paměť, klávesnice, myš, systémové hodiny
  + schránka
    - pro výměnu dat mezi procesy (ve formě textové, grafické, ale také operace)
* **struktura programů (aplikací) v MS Windows** 
  + je obdobná jako v systémech OS/2, Apple MacIntosh, NextStep a další
  + charakteristickou vlastností těchto programů je schopnost reagovat na události
    - vnitřní, nebo přicházející od uživatele
    - jedná se o „EVENT DRIVEN" programy
    - takovou událostí může být stlačení klávesy, posun myší, ale také zpráva od jiného programu
    - z hlediska architektury se pak uživatelský program více blíží operačnímu systému jako dosud známým programům
  + nevýhodou jsou vysoké nároky na technické vybavení, z hlediska programátorského vysoké nároky na způsob myšlení a také vytváření programů
  + nesplnění požadavku výkonného hardware může mít za důsledek nepřiměřeně dlouhé časové odezvy na vzniklé události
* **principy**
  + aplikační programy pod MS DOS v případě potřeby volají služby BIOSu, nebo přímo MS DOSu
    - mnoho aplikací je dokonce ignoruje a obrací se na technické prostředky přímo
    - protože ale BIOS a jádro MS DOS nejsou programy re-entrantní (s možností vícenásobného vstupu do té jisté procedury v případě vzniklého přerušení), je možné v jednu chvíli vykonávat pouze jeden program
    - v případě Windows se všechna volání služeb BIOSu a DOSu (INT 21H) zachytí a přesměrují do Windows
    - zde se zařadí do FRONTY UDÁLOSTÍ a postupně jsou zpracovány
  + filozofie systému je postavena na programových objektech
  + jejich základní jednotkou je okno
  + z hlediska programového se jedná o objekt s určitými vlastnostmi (atributy) a funkcemi (metodami), které určují jeho chování
  + veškerá komunikace s oknem probíhá systémem zasílaní zpráv (messages)
  + zprávy jsou po svém vzniku ukládané do posloupnosti dat zvané fronta (QUEUE)
  + systém MSW má k dispozici 1 systémovou frontu a pro každý běžící program jeho vlastní frontu programovou
  + po vzniku vnější události způsobí příslušný ovladač přerušení a vykoná se funkce MSW, která transformuje událost do tvaru zprávy s informaci o této události
  + zpráva se uloží do systémové fronty a čeká na své zpracování
  + současně je však systémem MSW vykonávaná funkce, která z opačného konce frontu zprávy odebírá a zpracovává
  + systém nejdříve rozhodne, kterému z aktivních procesů budou sdílené zdroje přiděleny, a pak je zpráva vložena do programové fronty odpovídajícího procesu
  + v průběhu tohoto zasílaní zpráv může systém řídit také přepínaní jednotlivých procesů (tedy přidělování centrálního procesoru)
  + v systémech s přidělováním času (tzv. **preemptivní multitasking**) se toto děje nezávisle na procesech

## SUBSYTémy

* Součást prostředí pro obyčejné aplikace, obecně dobře dokumentováno
* Subsystém **Windows**
  + Základní komponenty:
    - hlavní proces subsystému **csrss.exe** a knihovny DLL, které používá
    - ovladač jádra win32k.sys (drivery grafické karty a videa)
    - **vrstva knihoven DLL**
    - zajišťují překlad volání dokumentovaných funkcí Windows API a nativní volání rutin z knihovny ntdll.dll odpovědná za volání jádra (kernel32.dll, user32.dll, gdig32.dll, advapi32.dll)
    - csrss.exe v sobě uchovává vlastní kopii seznamu běžících procesů a vláken
* Subsystém **POSIX**
  + Portable operating systém interface based on Unix
  + sada mezinárodních standardů, popisuje aplikační rozhraní v OS založených na Unix
  + od Windows Server 2008 implementace v podobě SUA (Subsystem for Unix-based Application)
  + spouští se jen v případě potřeby
  + hlavním procesem – psxss.exe (komponenta subsystému běžící v režimu jádra – win32k.sys)

## SYSTÉMOVÉ PROCESY

* Proces **System**
  + PID=4
  + nevykonává řádný kód v uživatelském režimu
  + nepoužívá žádné knihovny DLL
  + neprezentuje žádný soubor \*.exe
  + v jeho kontextu běží skupina vláken - tzv. pracovní vlákna
    - vytvořena během bootovacího procesu
    - vykonávají činnost, kterou dostanou zadánu z venku
* Session manager = **smss.exe**
  + spouštěn v poslední fázi startu systému, jedná se o první proces vykonávající kód v uživatelském režimu
  + provádí poslední inicializaci systému, pak je systém připraven k přihlášení uživatele
  + hlavní úkol – vytvářet relace
    - relace se využívají pro oddělení prostoru jednotlivých uživatelů
    - smss.exe vytvoří relaci 0 tzv. konzolovou relaci
    - v ní běží systémové procesy, služby, všechny procesy prvního přihlášeného uživatele
    - každý další uživatel má svoji relaci
    - pro každou novou relaci správce spustí kopii **winlogon.exe**
    - pro relaci 0 spouštěn **wininit.exe** místo winlogon
  + smss.exe zodpovědný za inicializaci hlavního procesu subsystému Windows – **csrss.exe**
* **winlogon.exe** & **LSASS.exe**
  + Procesy podílející se na přihlašování uživatele
  + **winlogon.exe** umožňuje uživateli přihlášení pomocí grafického uživatelského rozhraní
    - zabudovaná jednoduchá ochrana proti zachycení přihlašovacích údajů
    - zajišťuje odhlášení uživatele
    - po zadání jména a hesla údaje odeslány **lsass.exe**
  + **LSASS.exe** zajišťuje ověření
    - zjistí oprávnění uživatele
    - vytváří token, kterým se uživatel dále prokazuje při provádění operací
    - obdrží-li winlogon token od LSASS, zahajuje inicializaci pracovního prostředí (spuštění startovacích skriptů, předání řízení userinit.exe)
* **services.exe** & **svchost.exe**
  + Procesy pro podporu služeb
  + ve Windows mnoho součástí jako služby
  + programy na pozadí nevyžadující interakci s uživatelem
  + mezi služby řazeny i ovladače jádra
  + služby mohou sdílet virtuální adresový prostor jednoho procesu
  + služby bez vlastního procesu vykonávány v kontextu instancí procesu svchost.exe
  + kontrola a správa skrze **services.msc**

verze Windows

* **klientské**
  + **Windows 95**
    - systém se snažil ulehčit práci se soubory a obrázky a dával systému
    - problémy nastaly s instalací jakýkoliv driverů, musely se instalovat každý zvlášť
    - používaly výhradně DOS
      * DOS byl nedílnou součásti systému
      * bylo potřeba zavést *autoexec.bat* a *config.sys* pro zavedení ovladačů atd.
    - instalace se vešla na asi 10 disket
  + **Windows 98**
    - nepotřebovaly k instalaci už DOS
    - instalace probíhá o něco déle
    - 98čky se používaly ve formě druhého vydání, tzn. Windows 98 SE
      * opravily některé chyby a nedostatky
  + **Windows ME**
    - neúspěch
    - nefungovaly některé hry
    - poslední verze založená na MS-DOS
  + **Windows XP**
    - rozdíl mezi Windows 2000 a XP se dá shrnout do pár bodů
    - kosmetické změny
    - určen pro domácí počítače (první domácí verze s NT jádrem) a pro podnikové sítě
  + **Windows Vista**
    - mnoho edic
    - často zatracován
    - relativně velké hardwarové nároky
    - nové grafické rozhraní Aero
    - nelze spustit na každém počítači
    - vyšší zabezpečení pomocí technologie UAC
      * uživatel není ve výchozím stavu administrátor, nutno práva případně elevovat
  + **Windows 7**
    - lehká evoluce vzhledem k Windows Vista
    - nové funkce
    - upravené grafické rozhraní
    - vyladěná technologie UAC
      * uživatel není moc často obtěžován výzvami k elevaci práv
    - vylepšený výkon, nižší hardwarové nároky
  + **Windows 8** & **8.1**
    - Prostředí Metro
    - Významná evoluce jádra, lepší optimalizace
    - úprava technologie UAC a administrátorských práv – skutečný administrátor je jen profil „Administrator“, uživatel s maximálními právy stále musí elevovat svá práva při některých operacích (lze obejít nastavením v secpol.msc nebo registrech, ale je tím omezena funkcionalita některých systémových aplikací)
* **serverové**
  + **Windows NT**
    - první verze založená na novém bezpečnějším jádře NT (New Technology)
    - určen pro podnikové sítě
  + **Windows 2000**
    - velmi stabilní
    - používá úplně jiný systém zabezpečení dat
    - určen výhradně pro sítovou konfiguraci a práci v síti
    - vyžaduje lepší procesor a dostatečné místo na disku
  + **Windows Server 2003**
    - Společní jádro s Windows XP
  + **Windows Server 2008**
    - Společné jádro s Windows 7

# Operační systémY Linux

Základní myšlenky, výhody a nevýhody, open-source, linux a GNU, start systému, moderní architektura jádra, jádro a vnější rozhraní jádra, základy ovládání - shell

## Základní myšlenka

* poskytnout lepší než plnohodnotnou náhradu drahým komerčním produktům
* open-source
  + počítačový software s otevřeným zdrojovým kódem
  + otevřenost zde znamená jak technickou dostupnost kódu, tak legální dostupnost - licenci software, která umožňuje, při dodržení jistých podmínek
* uživatelům zdrojový kód využívat, například prohlížet a upravovat

## Výhody a nevýhody

* **výhody**
  + open-source/BSD
    - zdarma k použití libovolně jak na stanicích, tak na serverech
  + velmi slušná dokumentace
  + silná podpora komunity na internetu
  + velké množství rozšiřujících balíčků
  + možnost vlastních úprav
* **nevýhody**
  + malé povědomí uživatelů (hlavně na pracovních stanicích)
  + obtížnější konfigurace pro správnou plnohodnotnou funkčnost
  + výrazně menší podpora výrobců ovladačů HW
  + menší množství známého zaběhlého SW spustitelného pod těmito systémy
  + u některých (MacOS X) úzká souvislost s danou platformou

## licence

* **komerční**
  + placená verze
  + demoverze
    - ukázková verze s omezenými funkcemi
  + shareware
    - verze k vyzkoušení
* **trial verze** 
  + časově omezená verze na vyzkoušení
* **freeware** 
  + software k užívání zdarma
* **donationware**
  + zdarma s žádostí o příspěvek autorovi
* **adware**
  + programy s integrovanou reklamou
* **public domain** 
  + volné dílo po všech stránkách
* **open source** 
  + volné dílo s volným zdrojovým kódem
* **EULA**
  + instalací uživatel „podepisuje“ smlouvu uvedenou při instalaci
* **BSD**
  + svobodný software, vyžaduje pouze uvedení autora
* ***GNU General Public License (GPL)*** 
  + volné, ale vyžaduje, aby dílo odvozené z tohoto díla bylo rovněž pod touto licencí (tzv. copyleft)

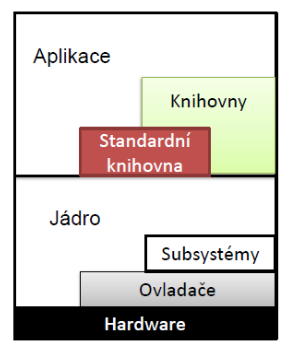
## Unix

* v informatice ochranná známka operačního systému vytvořeného v roce 1969 v Bellových laboratořích americké firmy AT&T
* v současné době existuje mnoho variant původního Unixu, které vyvinula AT&T a další organizace
* operační systémy pro servery, pracovní stanice a v současné době i pro osobní počítače
* hrály velmi výraznou roli při vzniku Internetu a přechodu od jednotlivých počítačů k počítačovým sítím a modelu klient-server
* naprogramován v jazyce C
* Unix je synonymem pro otevřený systém
* **filozofie**
  + jednoduchý
  + víceúlohový (multitasking)
  + víceuživatelský
    - uživatelé s domácími adresáři
    - individuální konfigurační soubory
    - přístupová oprávnění
    - současná práce více uživatelů
* hierarchický souborový systém
  + strom adresářů s jedním kořenem
* téměř vše je soubor
  + zařízení a prvky meziprocesové komunikace
* data (konfigurační soubory) jsou uloženy jako holý text
* důraz je kladen více na vztahy mezi programy, než na programy samotné
* sada jednoúčelových nástrojů, které dobře plní svůj specifický úkol
* propojování nástrojů do kolon (výstupu programu je přesměrován do vstupu dalšího)
* využívání hotových programů v jiných programech
* je orientovaný na zpracování textů
* je složen z
  + jádra operačního systému
  + systémových nástrojů (tzv. utility)
  + dalších aplikací15
* z UNIXu vycházejí systémy
  + SolarisOS od Sunu
  + MacOS X od Apple
  + částečně Linux

## Linux

* monolitické jádro operačního systému šířeného pod licencí GNU GPL v2
  + **flexibilní** – jedna verze na různém HW
  + **moderní** – rychlé nasazení nejnovějších technologií, protokolů, algoritmů atd.
  + **evoluční** – není řízen pevně definovaným návrhem, ale na základě požadavků
  + **svobodný** – volný přístup ke kódu jádra
* je známým příkladem svobodného softwaru a vývoje open source softwaru
* **patří mezi unix systémy**
  + vyhovuje standardy POSIX
  + vznikl jako svobodná implementace jádra unix systémů
* co ho odlišuje?
  + vnitřní architektura jednotlivých jader
  + univerzální směrování
  + domovská platforma x86
* ačkoliv termín Linux značí Linuxové jádro, často se používá pro označení celých unixových operačních systémů, které sestávají z Linuxového jádra a zároveň z knihoven a nástrojů z projektu GNU, ale i z dalších zdrojů
  + => ***GNU/Linux***
* podpora pro využití na serverech, a poslední dobou získává popularitu i na desktopovém trhu
* v roce 1991 započal vývoj jádra, které nakonec dostalo jméno „Linux“
* původně ho začal psát finský student helsinské univerzity **Linus Torvalds**
* vcházel z Minixu, což byl zjednodušený klon Unixu
* první verze linuxového jádra (0.01) byla vydána na Internetu 17. září 1991
* linuxová distribuce uceleně spojuje základní systém s velkým balíkem aplikačního softwaru, a navíc často zajišťuje uživatelsky přívětivou instalaci a následné aktualizace
* různé distribuce byly vyvinuty k různým účelům
  + možnost mít hotový systém připravený k použití, lokalizace, podpora určité počítačové architektury, použití v real-time a v embedded systémech
* v současné době existuje kolem 450 různých distribucí
* typická distribuce k obecnému použití obsahuje
  + linuxové jádro
  + určité GNU knihovny a nástroje
  + příkazové shelly
  + tisíce balíčků aplikačního softwaru
* **nejznámější distribuce**
  + Debian
  + Fedora
  + Red Hat
  + Gentoo, Knoppix, Mandriva, SuSE, Ubuntu

## JÁDRO LINUXU

* **Kernel**
* Hlavní úkoly jádra:
  + přidělovat paměť
  + čas procesoru (či procesorům)
  + čas programům
  + ovládání zařízení počítače (pomocí ovladačů) a abstrakci funkcí (aby bylo např. možné načítat soubory z pevného
  + disku a z jednotky CD-ROM stejným příkazem)
* Linuxové jádro je koncipováno jako jednolitá část kódu s podporou načítání externích modulů (tj. **modulární monolitické jádro**)
* používáno pro:
  + zvýšení stability
  + urychlení běhu jádra
  + zmenšení velikosti samotného jádra
  + zmenšení paměťových nároků
* tím zajištěna možnost připojování a odpojování jednotlivých modulů či ovladačů za běhu systému
* Zavádění systémů má dvě základní charakteristiky:
  + **dynamic linking** – moduly jádra mohou být zaváděny do kernelu, zatímco kernel zpracovává jiné úlohy
  + **stackable modules** – moduly jsou uspořádány hierarchicky, jednotlivé moduly jsou nadřazenými využívány jako knihovny pro přístup k nižším
* Linuxové jádro obsahuje podporu:
  + opravdového multitaskingu (umožňuje provozovat více úloh/aplikací zároveň)
  + virtuální paměti
  + správy paměti (správa paměti řízená jádrem nikoli aplikacemi)
  + sdílených knihoven a modulů
  + sdílených copy-on-write spustitelných souborů
  + nezávislých síťových vrstev podporujících mimo jiné síťové protokoly IPv4 a IPv6
* Základní pojmy na pomezí programů a jádra:
  + Uživatelský režim – systémový režim, v němž běží uživatelské procesy.
    - Lze využít jen omezenou instrukční sadu a omezení při práci s pamětí
  + Režim jádra – vykonává kód jádra.
    - Lze použít plnou instrukční sadu a neomezený přístup k systému
* Namapování režimů k režimům procesoru závidí na HW platformě.
  + Pro X86 máme 4 režimy (RING0 – RING3)
  + V převážné většině Linux využívá:
    - RING0 pro privilegovaný mód
    - RING3 pro neprivilegovaný mód
  + Při virtualizaci:
    - RING0 pro hypervizor
    - RING1 pro privilegovaný mód
    - RING3 pro neprivilegovaný mód
* Využití služeb jádra při implementaci aplikací – prostředky jádra:
  + Systémová volání
  + Signály
  + Sdílená paměť
  + Soubory a souborové systémy

## START systému

* Hardwarová kontrola systému (POST – Power On Self Test)
  + Spouští se firmware, který kontroluje:
    - paměť, základní součásti systému a periferie
    - Zde je možnost vstoupit do setup a nastavit další parametry, provádět další kontroly a detekce.
    - Na závěr se řízení předává BIOSu nebo LinuxBIOS. Ten má za úkol řídit další etapu startu.
* Načtení a start zavaděče
  + BIOS obsahuje základní funkce pro práci s HW. Podle seznamu zařízení začne hledat zavaděč systému (boot loader).
  + Zavaděč získá načtením startovacího záznamu (boot record).
  + Dříve byl zavaděč uložen jen přímo v rámci startovacího záznamu.
  + Dnes jen hlavní část – natažení celého zavaděče odjinud.
  + Moderní zavaděče jako je GRUB nepotřebují znát konkrétní uložení jádra.
* Načtení a rozbalení jádra
  + Najde-li zavaděč jádro, musí se před spuštěním načíst a rozbalit do cílové paměti.
  + Rozbalení zajišťuje rutina na začátku jádra.
  + Nejčastější metoda komprese jádra je deflate.
  + Při rozbalování je kontrolována integrita jádra.
  + Příprava na inicializaci – příprava tabulky paměťových stránek a detekce procesoru.
* Inicializace jádra, základní detekce HW
  + Složitý proces
  + Nastavení přerušení, vytvoření datových struktur, inicializace ovladačů a detekce zařízení.
  + Ovladače v modulech nejsou inicializovány
  + Vytvoření speciálního vlákna, z něhož se stává první proces – init.
* Spouštění programu init
  + init má výhradní postavení – všechny uživatelské procesy vznikají právě pod ním
  + rodičem init je samotné jádro
  + Role init:
    - řízení startu systému
    - řízení a přepínání úrovní běhu (runlevels)
    - správa procesů, jejichž rodič skončil
    - řízení vypínání/restartu systém
* Postupné spouštění různých procesů
  + - Typické spouštění:
    - Připojení speciálních souborových systémů
    - Aktivace bezpečnostních technologií
    - Nastavení parametrů pro konzoly
    - Inicializace síťových rozhraní
    - Kontrola souborových systémů
    - Úklid v adresářích pro dočasné soubory
    - Zapnutí paketového filtru
    - Start základních systémových démonů
    - Start konzol, případně X Window system

## Shell

* je označení programu, který vytváří v počítači rozhraní pro uživatele
* shell umožňuje uživateli
  + využívat funkce jádra operačního systému
  + zejména spouštět programy
  + zajišťovat pro ně vstupy
  + zobrazovat uchovávat a přesměrovávat jejich výstupy
  + spojovat jednotlivé programy do kolon a podobně
* **shell má obvykle dva režimy**
  + první je **interaktivní režim**, ve kterém jsou vložené příkazy ihned zpracovány
  + druhý je **dávkový režim**, který provádí příkazy předem zapsané v textovém souboru.
    - v unixových systémech je označujeme shellové skripty
* shell příkazy interpretuje, a proto mu říkáme interpret příkazů
* **existují v podobě** 
  + **textové** (příkaz.řádek)
    - *bash, sh, csh, rc, ksh, zsh, command.com*
  + **grafické**
    - *dosshell, litestep, průzkumník windows, X-Window system (KDE, gnome, CDE)*

MacOS

* označení původního operačního systému pro počítače Macintosh firmy Apple
* plně nahrazen systémem MacOS X v r. 2001
* základ systému se jmenuje Darwin a je složen z hybridního unixového jádra XNU spolu s množstvím BSD, GNU a dalších open source nástrojů
* systém Mac OS X je plně přizpůsoben počítačům Macintosh
* aktuální verze 10.6 Snow Leopard (2009)
  + implementuje ZFS FS
  + Intel x64 architekturu aj.

BSD

* *Berkeley Software Distribution, též Berkeley Unix*
* odvozenina Unixu distribuovaná Kalifornskou univerzitou v Berkeley mající počátky v 70. letech 20. století
* jméno je rovněž společně používáno pro moderní následníky této distribuce
* společnost AT&T, v jejichž laboratořích byl Unix vyvinut, umožnila univerzitám ho poměrně výhodně získat
  + systém se tak rozšířil
* BSD verze implementovala mnohá rozšíření týkajících se všech částí systému
* mezi příznivci původního Unixu a BSD rozšíření byly později vedeny dlouhé spory (tzv. Unixové války)
* na počátku 90. letech byl vývoj ukončen a dosažené výsledky byly uvolněny pod liberální BSD licencí
* proti této verzi, 4.4 BSD byly následně vzneseny právní námitky, sporné části byly odstraněny a vydána verze 4.4 BSD-lite
  + na ní jsou založeny mnohé další odvozené systémy
* BSD bylo průkopníkem mnoha inovací moderního počítání
* Berkeley Unix byl prvním Unixem obsahujícím knihovny podporující zásobníky Internet Protokolu: Berkeleyské socket
  + integrací socketů s popisovači souborů unixového operačního systému se stalo téměř stejně tak jednoduchým číst a zapisovat data přes síť jako tomu bylo u přístupu k disku
* dnes jsou BSD systémy používány jako testovací hřiště pro technologii akademickými organizacemi stejně jako v mnoha komerčních a svobodných produktech
* celková kvalita jeho zdrojového kódu stejně jako jeho dokumentace (zejména referenční manuálové stránky, společně odkazované jako manové stránky) jej činí velmi vhodným pro mnoho účelů

Sun Solaris

* dříve označovaný jako SunOS
* operační systém UNIXového typu vyvinutý společností Sun Microsystems
* původně pro počítače používající architekturu SPARC
* vyznačuje se robustností a stabilitou
* dobře zvládá SMP konfigurace s velkým množstvím procesorů
* k dispozici jsou nejnovější technologie na monitorování a debuggování aplikací
* zdrojové kódy Solarisu byly až na malé výjimky uvolněny pod licencí CDDL
* projekt OpenSolaris byl zahájen 14. června 2005

# OPERAČNÍ SYSTÉMY LINUX – VNITŘNÍ STRUKTURA

Systémová volání, signály a reakce na signály, komunikace pomocí rour a zpráv, procesy a vlákna v linuxu, udev a HAL, správa paměti v linuxu

## Systémová volání

* **Volání jádra** – základní metoda komunikace aplikačních procesů s jádrem.
* Tvůrce programu je od systémových volání oddělen standardní knihovnou – nepotřebuje tedy znát přesné chování, je ale doporučeno ho znát.
* Uživatelský proces → jádro – dva oddělené světy.
  + Uživatelský proces běží ve svém adresním prostoru (instrukce i data) a vykonává kód programu.
  + Režim jádra má vlastní adresní prostor.
  + Systémové volání je z pohledu aplikačního programátora obyčejné zavolání funkce, z pohledu vývojáře jádra je to také zavolání funkce, ale uvnitř jádra.
* Konkrétní způsob volání záleží na procesoru (x86 – EAX)
* Jak se předá řízení jádru? – platformě velice závislá záležitost
  + Přerušení – starší i386
  + Speciální instrukce – x86
* Reakce na výsledek systém volání ukončí opuštěním jádra a předání výsledků do registru
  + Úspěšně zakončené volání, které nepřináší konkrétní hodnotu (vrací nulu)
  + Úspěšně zakončené volání, které vrací hodnotu, jakákoli nezáporná hodnota
  + Neúspěšně zakončené volání vrací zápornou hodnotu – kód chyby

## Signály

* nejstarší metody komunikace mezi procesem a jádrem a mezi samotnými procesy
* Princip:
  + Proces vykonává určitou činnost → přijde mu signál → přeruší původní činnost → obslouží signál → proces může pokračovat dál v práci
* Dělení podle implementace:
  + **Obyčejné signály**
    - bity v masce signálu
    - v příchozím signálu se odpovídající bit (určený číslem signálu) nastaví na jedničku
    - u zpracovávaného signálu se bit vynuluje.
  + **Real-time signály**
    - nazývané spolehlivé signály, používají frontu – zaručeno, že žádný signál není ztracen
    - real-time signály používají čísla od 32 výše
    - využívá se např. pro komunikaci mezi vlákny
* možnost předefinování reakce na daný signál
  + na většinu to možné je
  + výjimku tvoří
    - **SIGKILL** – okamžité ukončení procesu
    - **SIGSTOP** – zastavení procesu
* Dělení podle posílání signálů:
  + **Synchronní**
    - přesně víme, kdy ho proces obdrží.
  + **Asynchronní**
    - signál může přijít kdykoli za běhu procesu a reakce na něj by tomu měla být přizpůsobena.
  + U většiny signálů se vyskytují oba způsoby.
* Posílání a doručování signálů:
  + Dělení
    - signály posílané zásadně jádrem
    - signály posílané uživatelskými procesy
  + Základní metoda pro poslání signálu je využitím funkce **kill()**, ta umožňuje poslat signál jednomu procesu nebo všem procesům ve skupině (kromě init)
  + Existuje funkce **raise()**, která posílá signál stejnému procesu, který ji zavolal
  + Další speciální funkcí je **sigqueue()**, která je využívaná pro real-time signály a informuje, zda byl signál vložen do fronty
  + Často se využívá funkce **pthread\_kill()**, která slouží k poslání signálu určitému vláknu – lze použít jen v rámci jednoho procesu.
  + Lze zaslat signál i vláknům nebo skupině vláken různých procesů, k čemuž se využívá funkce **tkill()** a **tgkill()** – primárně určeny pro použití v knihovnách.
* Reakce na signály
  + Po obdržení signálu, musí proces na tento signál reagovat.
  + Je-li proces (vlákno) při doručení v jádře, vrátí se do uživatelského prostoru, zde vykoná obslužnou rutinu a vrací se zpět do jádra, odkud vzápětí vyskočí a návratovou hodnotou definovanou přerušeným voláním.
  + Je-li naopak proces v uživatelském prostoru, je donucen vstoupit do jádra. Zde proběhne ekvivalentní obsluha signálu. Rozdíl je v tom, že po návratu do jádra se pouze uklidí data pro obsluhu signálu a po přepnutí do uživatelského režimu proces dále pokračuje od místa, kde byl přerušen. (pokud však signálem nebyl ukončen)
  + Výjimkou je případ, pokud je signál doručen při obsluze výjimky. V takovém případě nevstupuje do jádra, signál je obsloužen na konci vyřízení výjimky a po obsloužení signálu proces pokračuje, jako kdyby žádný signál nepřišel.
  + Při normální situaci se při obsluze signálu používá zásobník příslušného vlákna. Na začátku obsluhy je v něm vytvořena struktura pro obsluhu signálu. Její součástí je i speciální volání **sigreturn()**, které má samostatný vstupní bod do jádra a zajišťuje návrat z obsluhy signálu a uklizení zásobníku.
  + Možné reakce:
    - Výchozí reakce – nezměníme-li předem reakci na signál, použije se ta, která je pro daný signál výchozí.
    - Ignorování – na příchozí signál se nebude nijak reagovat
    - Obslužná rutina – určí se funkce, která bude pro obsluhu zavolána
    - Synchronní zpracování – jedná se o zvláštní případ, ve skutečnosti je jen o to, že se v jádře čeká na příchod signálu a následně je vráceno jeho číslo.
* Blokace signálu
  + Existují případy, kdy nechceme, aby signál byl doručen okamžitě, ale až v určitou chvíli.
  + K manipulaci s blokacemi signálu se využívá jejich bitová maska (datový typ **sigset\_t**) a sada funkcí.
    - Funkce **sigemptyset()** masku signálu vynuluje – nebude tedy obsahovat žádné signály.
    - Funkce **sigfillset()** ji naopak naplní, **sigaddset()** přidá signál do masky, **sigdelset()** signál z masky odstraní. Pro kontrolu, zda je určitý signál v masce používáme **sigismember()**.

## Roury a zprávy

* Metoda místní komunikace mezi procesy.
* Dělení:
  + Anonymní roury
  + Pojmenované roury
  + Komunikace pomocí zpráv
* **Anonymní roury (pipe)**
  + Nejjednodušší mechanismus pro komunikaci mezi procesy
  + Jedná se o proud bajtů – do jednoho konce se zapisuje, z druhého se čte
  + Na obou koncích může být ten samý proces nebo různé procesy
  + Konec roury může být sdílen i více procesy
  + Z pohledu implementace jde o rozhraní k paměťovému bufferu
  + Zapisuje-li do roury několik procesů zároveň, nelze garantovat pořadí, ve kterém se data v rouře objeví
  + **Komunikace s podprocesy**
    - Vytvořit jich dostatečný počet a nechat je zdědit
    - Chceme-li komunikovat s procesem, který o nich „nic“ neví, musíme jimi nahradit standardní výstup, vstup nebo chybový výstup
    - Náhrada proběhne po vytvoření podprocesu, ale před spuštěním nového programu
  + **Komunikace mezi vlákny**
    - Vytvoří se roura, jejíž jeden konec se používá v jednom vlákně a druhý v jiném, nebo i ve více vláknech
    - Při rušení vlákna je doporučeno zavřít příslušný konec roury
* **Pojmenovaná roura (named pipe, fronta FIFO)**
  + Komunikační objekt používaný podobně jako nepojmenovaná roura
  + Pojmenovaná roura přebývá ve svém domovském adresáři i v době, kdy s ní žádný proces nepracuje
  + Může být tedy uložena do archivu a následně obnovena – jde vlastně o speciální soubor
  + Před využitím pojmenované roury se musí vytvořit, k tomu využijeme volání **mkfifo()**
  + Jako argument využívá cestu v souborové systému a masku oprávnění
  + Pokud rouru nepotřebujeme, zrušíme ji voláním funkce **unlink()**
* **Komunikace pomocí zpráv**
  + Roury jsou objekty orientované na proud bajtů – nestrukturovaná data
  + Potřebujeme-li přenášet zprávy o pevně daném formátu, využijeme technologii určenou pro tuto oblast např. POSIX Message Queues
    - Technologie specifikovaná přímo na POSIX
    - Vyžaduje podporu jádra, příslušnou knihovnou pro uživatelské programy
    - Technologie POSIX PMQ umožňuje vytvářet pojmenované fronty\*
    - Do téže fronty může proces zprávy posílat a odebírat
    - Notifikace při příchodu zprávy může být asynchronní nebo synchronní
    - Každá zpráva má určenu prioritu
    - Linux podporuje 32 768 úrovní priorit
    - Vytvořená fronta funguje do restartu systému nebo jejího zrušení
    - Nepřečtená zprávy zůstávají ve frontě

## Procesy a vlákna

* Základním úkolem jádra je správa běžících procesů a vláken:
  + vytváření
  + plánování
  + nastavování
  + ukončování
* **Proces** – běžící program
  + Objekt, který pracuje podle kódu program (včetně knihoven), využívá svůj vlastní adresní paměťový prostor. Využívá služby jádra a komunikuje různými způsoby s ostatními procesy
  + Každý proces má jednoznačnou číselnou identifikaci označenou zkratkou PID (process idenfifiter).
  + Každý proces má svého rodiče – proces, kterým byl vytvořen.
  + Jedinou výjimkou je proces **init**
  + Životní cyklus:
    - Běžící
    - Zastavený
    - Pozastavený – přerušen kvůli nesplnění nějaké podmínky, například debuggerem
    - Zombie – provádění kódu bylo dokončeno, čeká na odstranění rodičovským
* **Vlákno** – objekt pracující podle kódu programu
  + Liší se od procesu tím, že je jeho součástí a sdílí prostředky (adresní prostor paměti, prostředky jádra atd.) s ostatními vlákny procesu
  + Hlavní vlákno – vlákno, které je spuštěno v procesu jako první – nepotřebuje explicitně žádnou obsluhu
  + Každé vlákno označeno identifikátorem TID (thread identifier) přidělované ze stejného číselného prostoru jako PID (TID hlavního vlákna = PID procesu)
  + Životní cyklus:
    - Běžící – nějaký procesor právě vykonává kód vlákna
    - Připravené k běhu – vlákno čeká na časové kvantum pro svůj běh
    - Přerušitelně uspané – vlákno čeká na splnění nějaké podmínky – lze ho probudit, takže např. může obsloužit signál
    - Nepřerušitelně uspané – vlákno čeká na splnění podmínky, ale nelze ho probudit
    - Zastavené – vlákno bylo zastaveno signálem
    - Pozastavené – opět podobná situace jako u procesu
    - Skončené – vlákno doběhlo do konce a čeká se, až se s ním vlákno spojí
  + S vlákny se obvykle pracuje prostřednictvím knihovny libpthread, která hlavně implementuje rozhraní POSIX Threads
  + Důležitou funkcí při ukončování vlákna je zavolat tzv. úklidovou obslužnou funkci (clean-up handler)
* **Úloha (task)** – objekt, jehož kód se sekvenčně vykonává
  + Každému vláknu procesu odpovídá jedna úloha (kromě toho může proces využívat ještě další, navenek neviditelné, úlohy přímo v jádře – např. vlákna jádra spravující pracovní fronty.)
  + Na jednom procesoru běží v určitý okamžik nejvýše jedna úloha
  + Úlohy běží na jednotlivých procesorech v rámci přidělených časových kvant.
* **Plánování úloh**
  + Jak bude plánování vypadat, ovlivňují tři faktory:
    - Použitý plánovač
    - Priorita úlohy
    - Vlastní jádro (preemptivita)
  + Každé vlákno procesu je z hlediska plánování chápáno jako samostatná úloha, je tedy plánováno nezávisle na jiných vláknech téhož procesu.
  + Plánovač má nejzásadnější vliv na plánování. V jádře jsou dnes implementovány čtyři plánovače.
    - Normální plánovač – pro většinu použití, časová kvanta přiděluje na základě dynamických priorit, které jsou určovány podle toho, jak úloha v minulosti využívala procesor.
    - Dávkový plánovač – jako normální plánovač, ale s úpravou pro neinteraktivní úlohy. Liší se výpočtem dynamických priorit, na základě které se určuje velikost časového kvanta.
    - Plánovač FIFO – pro úlohy zpracovávané v reálném čase. Tento plánovač funguje tak, že úloze přidělí procesor, a dokud ho on sám neodevzdá, žádná jiná úloha na tomto procesoru příležitost k běhu nedostane. Využívá 99 úrovní statických priorit.
    - Plánovač round-robin – podobný FIFO, liší se přidělováním časových kvant, které přiděluje dokola na určité úrovni priority (úrovní 99).
* **Preemptivita jádra**
  + Nepreemptivní chování – úloha se v jádře musí sama vzdát procesoru
  + Dobrovolně preemptivní chování – jsou přidány body, kde se úloha vzdá procesoru
  + Plně preemptivní chování – plánovač úloze odebere procesor, pokud vyčerpala časové kvantum

## UDEV& hal

* **udev**
  + Jde o virtuální zařízení představující rozhraní mezi uživatelskými procesy a ovladači v jádře – ty se používají prostřednictvím speciálních souborů zařízení.
  + Tyto soubory se obvykle nacházení v adresáři /dev a jeho podadresářích velmi často se standardizovanými názvy
  + udev funguje jako proces, který běží v uživatelském prostoru. Nepotřebuje žádná zvláštní oprávnění – jen zápis do adresáře /dev a jeho podadresářů a přijímat informace o událostech hlášených do uživatelského prostoru
  + Pro svoji činnost udev nezbytně potřebuje souborový systém **sysfs**, odkud čerpá informace o virtuálních zařízeních. Při změnách v sysfs se musí také upravit implementace udev.
* **HAL**
  + Hardware Abstraction Layer je vrstva s cílem zajistit co nejabstraknější podobu viditelnosti fyzických zařízení – aplikační programy nemusí řešit technické detaily jednotlivých zařízení
  + Úkoly HAL
    - Načítání modulů s ovladači
    - Správa virtuálního hardwaru a pohledů na něj
    - Komunikace s procesy ohledně změn ve stavu zařízení
    - Vytváření přípojných bodů, připojování souborových systémů
* Udev a HAL spolu komunikují skrze sběrnici **D-BUS**

## Správa paměti

* **Hlavní úkoly**
  + Převody mezi různými druhy adres
  + Zpřístupnění paměti, která leží mimo adresní prostor
  + Nízkoůrovňová alokace pro různé účely
  + Správa paměti a alokace pro jádro
  + Správa paměti procesů, přidělování, mapování, sdílení
  + Různé cache, přednačítání, zpožděný zápis
  + Odkládání na disk, zpětné načítání
  + Řešení nedostatku paměti
* **Zóny paměti**
  + Z důvodů HW limitů je paměť rozdělena na zóny, kde každá má své vlastnosti předurčující její použití
  + Dělení na zóny je dáno architekturou – liší se u různých architektur – obecně:
    - Zóna pro DMA
    - Zóna pro normální použití
    - Zóna mimo adresní prostor
* Alokační **algoritmus buddy**
  + Skupiny bloků od 1 do 1024 stránek
  + Vždy zdvojnásobuje další úrovně velikosti
  + Celkem 11 úrovní velikosti
  + Pravidla pro alokaci:
    - Zkus najít blok odpovídající velikosti – pokud existuje – přiděl ho
    - Zkus najít blok o úroveň větší velikosti – neexistuje-li, pokračuj na další úroveň
    - Nalezený blok rozděl tak, aby se přidělila potřebná část a zbytek byl použitelný pro další zpracování
* Často se alokují jednotlivé stránky – využití buddy pomalé
  + Pro každý procesor v jádře vytvořeny **cache** – horká (hot cache) a studená (cold cache)
* Není jednoduché nalézt souvislý úsek paměťových stránek – alokace nesouvislých úseků a jejich namapování
  + Alokuje se sada úseků a ty se namapují do virtuálního adresného prostoru jádra – získáme souvisle adresovatelný úsek – pak lze normálně pracovat
* Metody řešení nedostatku fyzické paměti:
  + Uvolnění aktuálně nevyužívaných stránek
  + Aktivace mechanismu **OOM killer** – postupně ukončí tolik procesů, aby vyřešil situaci – prochází všechny procesy a vybere ten s nejvyšším skóre (**badness**)
  + Kategorie uvolňovaných stránek:
    - Odložitelné
    - Synchronizovatelné
    - Přímo uvolnitelné
* Reverzní mapování
  + Používá se pro najití všech položek odkazujících na určitou stránku
  + V linuxovém jádře se používá objektové reverzní mapování a to buď anonymní, nebo souborové

# Serverové operační systémy

specifika serverových operačních systémů, rozdíly mezi OS pro osobní počítač a pro server, serverové služby, správa uživatelů, role serverů, zálohování dat včetně RAID

## server

* + obecné označení pro počítač, který poskytuje nějaké služby nebo počítačový program, který tyto služby realizuje
  + **služby**
    - služby server poskytuje klientům, což označujeme jako model klient-server
    - poskytování služby zajišťuje speciální program
      * v unixových systémech je označován jako démon (anglicky *daemon*)
      * v Microsoft Windows pak jako služba (anglicky *service*), která s klientem komunikuje pomocí definovaného protokolu
    - dedikovaný server
      * vyhrazený pro speciální účely, bez přímého přístupu uživatelů
  + **hardware**
    - výkonnější, než jsou obyčejné stanice pro uživatele
    - mohou obsahovat záložní (redundantní) komponenty (dva zdroje, diskové pole RAID, …)
    - nepotřebují grafické a zvukové karty, nezbytné nejsou ani monitory
    - potřebuje hodně paměti, diskového prostoru, I/O rozhraní, síťového rozhraní
    - lepší chlazení
    - UPS záložní zdroje
    - RAID disková pole
    - ECC paměti s kontrolou parity a schopností opravit chyby
    - často umístěny v racku, v serverové místnosti
    - možnost instalovat víceprocesorové jednotky (u velkých serverů)
    - zálohovací záznamová zařízení (magnetické pásky)
    - omezený přístup (fyzicky i softwarově)
  + **software**
    - speciální operační systémy (Windows) nebo standardní ale jinak nastavené (Linux)
    - škálovatelnost
    - nastavení přístupových práv
    - větší množství síťových služeb, monitoring, zálohování
    - neklade důraz na grafické rozhraní
    - většinou možnost změnit nastavení bez restartu

## Operační systém

* **serverový operační systém**
  + operační systém, který je určen pro běh na serverech
  + poskytuje speciální služby, především v oblasti
    - síťové správy
    - zálohování a organizace dat
    - poskytování síťových služeb
  + někteří výrobci OS vytvářejí speciální edice (MS)
  + jiní spoléhají na univerzálnost a rozšiřitelnost systému pro práci na serveru (Linux)
* **příklady**
  + *Linux*
  + *FreeBSD*
  + *(Open)Solaris*
  + *Microsoft Windows* 
    - *NT*
    - *2000*
    - *Server 2003*
    - *Server 2008*
* většinou se nasazují 64b verze kvůli podpoře většího množství paměti RAM (> 4GB)

## Rozdíly oproti OS pro stolní PC

* není nutné grafické rozhraní
* horší interaktivita s uživatelem
  + vyžaduje zkušenějšího uživatele
* podpora změny hardwaru (*hot-plug*) a softwaru bez nutnosti restartovat systém
* pokročilé služby v oblasti
  + zálohování
  + síťové správy
* podpora
  + diskových polí
  + víceprocesorových systémů
  + větší operační paměti
* vylepšené
  + bezpečnostní mechanismy
  + správa uživatelů, zdrojů
  + ochrana paměti a dat
* lepší podpora automaticky běžících služeb (démonů)
* mechanismy proti přehřátí procesoru, pamětí, selhání procesoru, disků atd.

## služby poskytované serverovým OS

* plnohodnotná správa uživatelů a uživatelských oprávnění
* služby pro zálohování dat, monitoring sítě, sdílení serverových prostředků (HW, SW)
* provozování služeb souvisejících s poskytováním internetových technologií (skripty, databáze)
* firewall a další bezpečnostní mechanismy
* zpřístupnění databáze/serverového disku pro uživatele
* odesílání e-mailů, FTP
* statistické služby, archivace

## role serverů

* **webový server** 
  + především v síti internet poskytuje www stránky
* **souborový server** 
  + slouží např. v podnikové síti jako centrální úložiště dat (dokumentů)
* **databázový server** 
  + slouží jako úložiště strukturovaných dat (databází)
* **tiskový server** 
  + zpřístupňuje počítačové tiskárny
* **síťový server** 
  + DNS, DHCP, e-mail…
* **faxový server**
* **proxy server** 
  + zprostředkovává přístup do jiné sítě jiné (např. internet)
  + cache…
* **aplikační server** 
  + počítač specializovaný na provoz nějaké aplikace
* **herní server** 
  + nabízí hraní her s více hráči (multiplayer)
* **doménový server**
  + spravuje uživatelské účty v rámci domény *Active Directory* Windows

## Zálohování dat

* speciální software, někdy i hardware (pásková jednotka)
* **média**
  + pásky
  + CD, DVD
  + RAID pole
* nutno zálohovat v pravidelných intervalech
* zálohy nutno uchovávat odděleně od serveru (jiná budova, místnost)
* **typy záloh**
  + **úplná – inkrementální** 
    - zálohuje celek, pak pouze ty soubory, co se změnily
  + **úplná – rozdílová**
    - podobně jako inkrementální
    - akorát zálohuje i všechny vytvořené soubory
  + **zrcadlová – reverzní** 
    - ukládá historii změn
* velká data možno komprimovat

## RAID pole

* redundantní pole nezávislých disků
* typ diskových řadičů, které zabezpečují pomocí určitých speciálních funkcí koordinovanou práci dvou nebo více fyzických diskových jednotek
* zvyšuje se tak výkon a odolnost vůči chybám nebo ztrátě dat
* RAID se používá výhradně v souvislosti s rozhraním SCSI (současný zápis na víc disků)
  + existují sice i základní desky, které obsahují EIDE řadič s podporou RAID na hardwarové úrovni, ale je to jen takové hraní na desktopech, pro servery pouze SCSI
* prakticky se používají tři typy
  + RAID 0
  + RAID 1
  + RAID 5
  + příp. kombinace RAID 1+0
* **RAID 0**
  + řetězení
    - ukládání dat na více disků
    - bez možnosti obnovit po ztrátě
  + prokládání
    - rozdělení souborů na menší celky a ukládání celků střídavě na všechny disky
    - zrychlení přístupu k datům
  + obě metody vytvářejí velký logický disk spojující data na fyzických discích
* **RAID 1**
  + zrcadlení
    - obsah je současně ukládán na dva stejné disky
    - vypadne-li jeden, nedojde ke ztrátě žádných dat
    - opačná alternativa k RAID 0
    - neřeší rychlost, ale pouze bezpečnost – chrání před výpadky systému
    - 50% diskové kapacity běžně nevyužito (jen záloha)
    - u více disků nevýhodné, navíc neřeší rychlost
  + duplexing
    - obdoba zrcadlení, jen jsou použity nejen dva disky, ale i dva samostatné řadiče disků
    - zvyšují odolnost
  + stripping 1+0, 0+1
    - kombinace RAID 0 a 1
    - vyžaduje min. čtyři disky
  + RAID 0 a 1 nevhodný pro databáze
    - příliš časté manipulace s transakčním souborem (mnoho přístupů – malý výkon)
* **RAID 2**
  + obdoba RAID 1 s rozdílem, že už disky nejsou připojeny k jednomu řadiči, ale systém je zdvojený (2 SCSI sběrnice)
    - zvlášť jsou live data a zvlášť backup
  + řeší navíc i problém výpadku jednoho z řadičů (systém nepadne)
* **RAID 3**
  + vůbec se neujal
* **RAID 4**
  + „*data stripping with* *parity*”
  + první diskové pole řešící problém rychlosti a současně i bezpečnosti
  + použití s min. 3 disky (u předchozích stačily 2)
  + 1. a 2. disk s live daty a 3. disk paritní
  + na paritní disk jsou ukládány informace, s jejichž pomocí se v případě poruchy dopočítají ztracená data
  + nárůst rychlosti
  + v případě poruchy kteréhokoliv z disků systém zůstane funkční
  + při nákupu systému se používá strategie použít disky z různých výrobních sérií (aby se zamezilo vzniku stejných chyb na více discích současně
  + RAID 4 nefunguje optimálně při práci s malými soubory
  + zdržuje operace několikerého přepočítávání paritní informace v průběhu zápisu
* **RAID 5**
  + „*distributed parity*“
  + min. 3 disky
  + už neplatí, že je jeden disk výhradně paritní, ale systém paritní informace optimálně rozkládá po všech discích
    - zvýšení výkonu
  + výhody jako u RAID 4 + není brzděn při zápisu
  + disků může být mnoho, vyvstává však další problém s chlazením a současným náběhem disků (elektrický šok)