ARP 😫☠️

Architektura počítačů

- Architektura je návrh a konstrukce zařízení na zpracování dat; soubor pravidel popisujících funkčnost, organizaci a implementaci počítačových systémů
- Počítač je zařízení na číslicové zpracování informací; vrstvy abstrakce:
    - UI
    - Výkonné jádro aplikace
    - Aplikační knihovny
    - OS
    - Instrukční architektura (ISA)
    - Datová cesta, řízení
    - Logické obvody
    - Tranzistory
- Počítač = CPU (řadič CU + ALU) + IO + operační paměť + archivní paměť

Společné vlastnosti architektur

- Struktura počítače je nezávislá na typu řešené úlohy, počítač se programuje obsahem paměti
- Paměť je rozdělena do buněk stejné velikosti, jejichž očíslováním vznikají adresy
- Program je tvořen posloupností instrukcí (elementárních příkazů), které jsou jednotlivě prováděny v pořadí, v jakém jsou zapsány v paměti
    - Pro změnu pořadí lze využít instrukcí ne/podmíněného skoku
- Pro reprezentaci instrukcí i čísel se používájí dvojkové signály a dvojková číselná soustava
- Tok dat řídí řadič → programem řízené zpracování dat probíhá v počítači samočinně
- Zpracování dat probíhá v „diskrétním režimu“ → během výpočtu nelze s počítačem komunikovat

Harvard

- **Paměť programu je oddělena od paměti dat**
    - Možnost ve stejném okamžiku načítat instrukci a přistupovat k datové paměti
    - Datová a programová paměť mohou mít odlišnou organizaci
    - Programová paměť může být nevolatilní, program není nutné nahrávat z externího média
- Má oddělené sběrnice: datová, instrukční, adresová
- Řízení procesoru je odděleno od řízení vstupních a výstupních jednotek, nejsou napojeny přímo na ALU
- Rychlejší, vhodné pro zpracování většího objemu dat
- Použití zejména v mikrokontrolérech a signálových procesorech v kombinaci s RISC

Von Neumann

- **Instrukce a operandy jsou v téže paměti**
- Jedna sběrnice pro data, instrukce a adresy
- Vstupy jsou koncipovány jako datové zdroje a výstupy jako výsledky, jsou proto napojeny přímo na ALU
- Nevýhody:
    - Může být obtížnější ladění programu

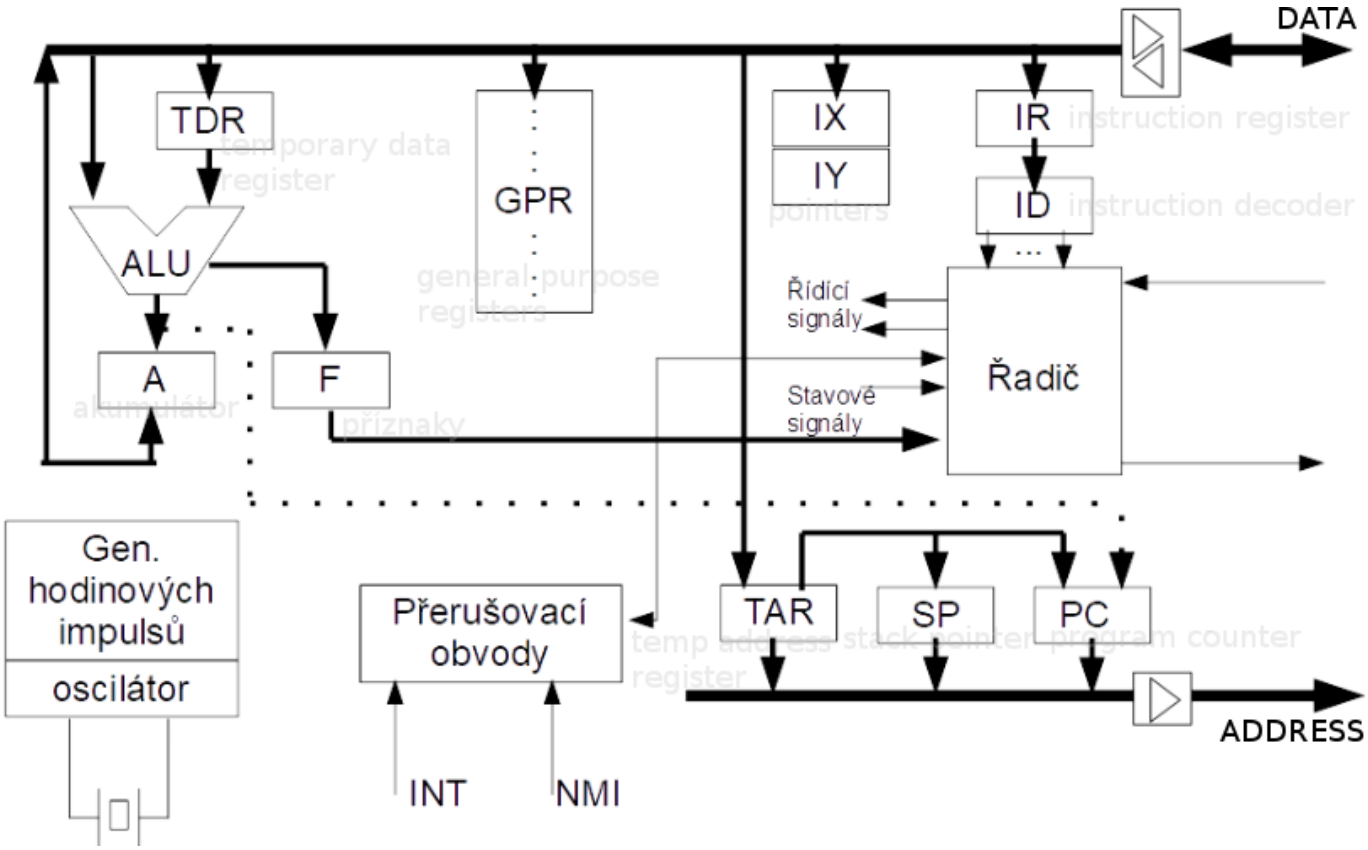
    - Rychlost pamětí je slabým článkem, proto CPU obvykle obsahují rychlou cache
    - Méně bezpečné; možnost mylně interpretovat data jako program
    - Data i instrukce se přenáší po stejné sběrnici

Výkonnost počítače

- Výkonnost je převrácená hodnota doby potřebné k vykonání jednoho úkonu
- Propustnost je množství vykonané práce za jednotku času
- CPI je průměrný počet taktů na instrukci
- Výkonnost CPU = (počet instrukcí × CPI × doba trvání jednoho taktu)$^{-1}$
- Měření výkonu:
    - Instrukční mix – seznam instrukcí ohodnocenými jejich četnostmi výskytu – nejstarší, závislé na CPU a konkrétní úloze/programátorovi, nebere v potaz režii OS
    - Zkušební úloha (benchmark) – vzorek zátěže v určité aplikační oblasti, výhodou je komplexnost: bere v potaz OS, překladač, IO, ...; dělíme na reálné a umělé
- Monitorování výkonu: programový nebo obvodový monitor

Procesory

- Procesor je základní jednotka počítače – automat pro zpracování informací obsahující CU a ALU; společně s IO a pamětí tvoří počítač; chování je definováno programem
- Dělení: univerzální, grafické, signálové, multimediální, šifrovací, AI, ...
- CPU obsahuje:
    - **CU – řadič**
        - Řídí chod celého procesoru podle instrukcí programu
        - Obsahuje instrukční registr a instrukční dekodér
        - IR uchovává opcode právě vykonávané instrukce
        - ID dekóduje instrukci a generuje řídící signály pro procesor
        - Existují dvě koncepce řadičů:
            1. **Obvodový řadič** – speciální sekvenční automat s čítačem a dekodérem; rychlejší, dražší
            2. **Mikroprogramový řadič** – řídící paměť s uloženými mikroinstrukcemi; adresový registr AR uchovává adresu registru pro r/w
        - CU dále obsahuje (pod)řadiče pro přerušení, IO, DMA, periferie, ...
    - **ALU – aritmeticko-logická jednotka**
        - Provádí všechny aritmetické a logické výpočty
        - Součástí bývá příznakový registr (carry, borrow, zero, negative, overflow, half-carry, parity, ...), použití pro větvení programu
        - Procesor může mít více ALU (např. specializované FPU)
    - **RS – sady registrů**
    - **PC – program counter** – často jeden registr z RS nebo součást CU
    - **Vnitřní sběrnice** – data, adresy, řízení



Instrukce

- Specifikace jednoduché činnosti, kterou má provést technický prostředek; soubor instrukcí tvoří program
- Binární tvar: opcode + operandy; obvykle zapisujeme v jazyce symbolických adres
    - Opcode: operační kód určující typ instrukce
    - Operandy (parametry): konstanty, označení zdrojových/cílových registrů, adresa paměti programu (pro skoky), ...
- Mohou se dále dělit na mikroinstrukce → odlišná externí a interní instrukční sada; lepší zachování zpětné kompatibility
    - Mikroinstrukce je obvykle uložena v jedné buňce ROM a její vykonání trvá jeden takt
    - Nanoinstrukce → dvouúrovňový paměťový systém
- Formáty kódování instrukcí:
    - Proměnná délka opcode – lepší hustota, náročnější dekódování (často v CISC)
    - Pevná délka opcode – rychlé a jednoduché dekódování, snazší implementace pipeliningu, delší programy (často v RISC)
- Instrukční sada – seznam instrukcí, datových typů, dostupných režimů, registrů, pravidel (adresování, přerušení, ...)
    - Dělíme podle: Koncept (ISA), implementace mikro/architektury (x86, ARM, ...), rozsah (CISC/RISC), míra paralelnosti (xSkalrání, VLIW)

ISA – Instruction Set Architecture

- Lze chápat jako rozhraní mezi HW a SW počítače
    - Souhrn vlastností počítačového systému z pohledu programátora v assembly language
- Definuje:
    - Způsob kódování instrukcí – pořadí operací/operandů/adres

    - Zacházení s operandem – způsoby adresování
    - Možné operace – aritmetické, logické, skoky, ...
    - Způsob ukládání výsledku – střadač / registr/ paměť / zásobník
    - Datové typy a velikosti operandů
    - Větvení – varianty skoků, volání a návrat z podprogramů, způsoby přerušení

Střadačově orientovaná ISA

- Střadač = akumulátor (speciální registr uchovávající hodnotu předchozí operace)
- Akumulátor je implicitním operandem aritmeticko-logických instrukcí (může být zdrojem i cílem)
- Druhý operand je explicitní (registr nebo paměťová buňka) → jednoadresní architektura
- Snadno kódovatelné instrukce (→ jednoduchý HW), krátké instrukce, rychlé přepínání kontextu, častý přístup do paměti, problematická realizace paralelismu mezi instrukcemi

Zásobníkově orientovaná ISA

- Vzácné, ale SW implementace se používá např. v JVM
- Bezadresní – instrukce PUSH a POP + operace ALU
- Zdrojem a cílem všech operací je HW zásobník
- Jednoduché a rychlé instrukce (bez operandů), vysoká hustota kódování, rychlá implementace
- Komplikovaný přístup k datům – nemožnost náhodného přístupu, obtížný paralelismus
- Řadič musí řešit přetečení/podtečení HW zásobníku

ISA s univerzálními registry

- Nejpoužívanější (x86), základem je soubor velmi rychlých univerzálních registrů (GPR)
    - Mohou být zdrojem i cílem (obsahují mezi/výsledky, proměnné, ...)
    - Kolem 8 až 128 registrů
- Instrukce kolem 2 až 3 operandů
- Registry jsou rychlejší než jakákoliv paměť, umožňují náhodný přístup, méně časté sahání do paměti (zrychlení), snadná implementace paralelismu
- Složitý překladač, registry nemohou uchovávat složené datové struktury (pole), přepnutí kontextu trvá delší dobu (ukládání více registrů)
- Některé registry mohou být „ne-univerzální“ (status, pointer, PC, ...)
- Varianty
    - `R-R` – oba operandy v registrech (jednoduché kódování, pevná délka instrukce, kostantní CPI, vyšší počet instrukcí pro daný program, typické pro RISC)
        - `Load/Store` – architektura s dvěma typy instrukcí: pro přístup k paměti a pro práci s daty (ALU)
    - `R-M` – jeden operand může být v paměti (přímý přístup k datům bez meziukládání, hustější kód, různé CPI, typické pro CISC)
    - `M-M` – oba operandy mohou být v paměti (zastaralé)

CISC – sada komplexních instrukcí

- Proč?
    - Dříve byly operační paměti značně pomalejší než CPU → zpomalování výpočtů opakovaným načítáním dat → snaha rozšířit instrukční soubor → mnoho složitých instrukcí používaných jen

zřídka
    - → Kratší programy (méně instrukcí) → úspora drahých pamětí a přístupů do pomalé RAM
    - Kompilátory byly méně dokonalé, používaly se jazyky na nižší úrovni → přesun složitosti ze SW do HW (řadiče) → menší rozdíl v úrovních abstrakce programovacích jazyků
- Typické vlastnosti:
    - Různá délka instrukcí; zpracování instrukcí ve více strojových cyklech (~5–10 CPI)
    - Velký počet adresovacích módů
    - Kvůli vysoké složitosti je řadič na principu ROM s mikroprogramy, ty je možné měnit
    - Řídící obvody (pro dekódování instrukce) zabírají na čipu velký prostor
    - Jednodušší překladač
    - Obvykle GPR ISA (`R-M`, dříve `M-M`)
    - Možnost zřetězení s rozkladem na mikroinstrukce

RISC – optimalizovaná sada instrukcí

- Snaha přesunout složité a málo používané CISC instrukce z HW do SW
- Relativně jednoduché instrukce s pevnou délkou (počet instrukcí není stěžejní faktor)
- Zvětšení počtu instrukcí v programu, ale celkové zrychlení (snížení CPI)
- Jednoduché instrukce umožňují vyšší frekvenci
- Pipelining
- Složitější kompilátory
- Řadič s pevnou logikou místo mikroprogramování (rychlejší)
- Řídící obvody zabírají méně místa
- Větší počet programově dostupných registrů, které jsou víceúčelové (jednodušší překladač) a lze je použít jako operand pro libovolnou instrukci
- Operace s daty pouze nad registry – 2 zdrojové a 1 cílový
- Malý počet adresových módů

Post-RISC

- Většina současných CPU – navenek CISC, uvnitř RISC-ish
- Instrukce trvají různou dobu, rozkládají se na mikroinstrukce, které podporují pipelining
- Spekulativní provádění instrukcí a zpracování instrukcí mimo pořadí
- Paralelismus
- Nadále dochází k rozšiřování instrukční sady (např. pro multimédia), obvykle zachovává zpětnou kompatibilitu

Instrukční cyklus

- Skupina všech dob (fází) nezbytných k uskutečnění jedné strojové instrukce
    1. Instruction fetch (IR ← mem[PC])
    2. Instruction decode (opcode → řídící signály pro ALU; ~PC++)
    3. Operand fetch
    4. Execute
    5. Memory operation (obsluha paměti/přerušení)
    6. Write back (zápis výsledku)

Pipelining (zřetězení)

- Kombinační (jednotaktový) CPU je model CPU, který uskuteční instruční cyklus v jednom taktu (CPI = 1)
- Vícetaktový CPU vkládá dodatečné registry k oddělení jednotlivých fází cyklu, některé fáze lze vynechat, nepatrně vyšší výkon, univerzálnější (mikroinstrukce), vyššího výkonu lze dosáhnout zřetězením
- Zřetězený CPU vkládá dodatečné registry mezi jednotlivé části řadiče
    - Zvýšení vytížení jednotlivých CPU komponent
    - Pro pipelining ideálně chceme nepřetržitý přísun instrukcí, které lze rozdělit na sekvence nezávislých kroků, jejichž trvání by mělo trvat podobnou dobu
    - Parametry: hloubka zřetězení (kolik úrovní), latence řetězu (doba průchodu jedné instrukce)
    - U velkého počtu fází vznikají prázdná místa (bubliny) → nižší výkon, zvyšování frekvence nutně nezvýší rychlost, ale určitě zvýší spotřebu
    - Pipeline vnitřně v CPU zavádí určitý paralelismus, zvenku se ale CPU tváří sekvenčně → mohou vznikat „hazardy" (narušení vykonávání programu)
        - Strukturní hazardy – kolize sdílených prostředků
        - Datové hazardy – instrukce čeká na jinou (na výsledek, na uvolnění registru, ...)
        - Řídící hazardy – ztráty při podmíněném skoku atd.
    - Řešení hazardů:
        - Statické při kompilaci – přeskládání instrukcí – vkládání NOPů nebo „zamíchání" s „nezávislými" instrukcemi
        - Dynamické za běhu – vkládání prázdných taktů řadičem
- CPU, instrukční cyklus a metody jeho zrychlení (stručně):
    - **Subskalární (sekvenční) procesory** – doba vykonávání programu je součet časů trvání jednotlivých instrukcí, klasika
    - **Skalární procesory** – jedna pipeline
    - **Superskalární procesory** – více nezávislých pipeline; dva druhy:
        - Statické (o paralelismu rozhoduje kompilátor, např. VLIW)
        - Dynamické (o paralelismu se rozhoduje za chodu, složitější, zachovává ale zpětnou kompatibilitu)
    - Superzřetězení – jednotlivé fáze zpracování instrukce jsou ještě dále děleny na kratší úseky
    - Fronta instrukcí – při delších instrukcích plníme frontu, abychom se pak nezdržovali načítáním instrukcí; při skoku nutno vyprázdnit
    - Out of order vykonávání instrukcí
    - Spekulativní vykonávání instrukcí
    - Register renaming; přidělení různých kopií stejného registru různým instrukcím
    - Vertikální multithreading – jedno aktivní vlákno; pokud na něco čeká, přepneme na jiné
    - Horizontální multithreading – superskalárnost z pohledu vláken (více vláken pracuje naráz)
    - Hyperthreading – do pipeline jednoho jádra se zavádějí nezávislé instrukce dvou vláken
        - Vlákna mají vlastní registry a PC, ale sdílené ALU a L2 cache
        - Dobrý poměr přidaného výkonu za dodatečně zabranou plochu

Další parametry/technologie procesorů

- Intel Turbo Boost – pokud to okolnosti dovolují (počet aktivních jader, teplota, očekávaná spotřeba), tak mohou jednotlivá jádra běžet na vyšší frekvenci (automatické zvýšení násobiče); užitečné při nerovnoměrné zátěži jader CPU; AMD má Turbo Core
- Intel Anti-Theft (placená služba, HW ochrana proti krádeži, zamčení na dálku, ...)

Komunikace CPU s IO zařízeními

- VV (vstupně-výstupní) brána je realizována IO portem nebo VV řadičem a slouží k předávání dat mezi počítačovou sběrnicí a periferií zařízení
- Periferie jsou pomalejší než CPU, realizace součinnosti:
    1. Přímá programová obsluha
        - Program ve smyčce testuje, která zařízení mohou odesílat/přijímat (vkládání čekacích taktů nebo využití handshake)
        - Pak je vyvolán VV podprogram (*ovladač – driver*), který zajišťuje řídící signály pro danou periferii
        - Zdržuje, vhodné pro rychlá zařízení
    2. Přerušení
        - (umět cotoje, skok&kontext, zdroje, priority)
    3. DMA – Direct Memory Access
        - Kopírovaní bloků dat mezi pamětí a portem bez průchodu přes CPU a bez dočasného ukládání těchto dat v pomocných registrech
        - Využití DMA řadičů (generuje adresy a řídící signály, žádá CPU o část paměti)
    4. Speciální VV procesor
        - Podobné DMA, ale více samostatné

Čísla s plovoucí řádovou čárkou

- Vychází z vědeckého zápisu čísel (scientific notation):
- $A = M \cdot 2^E$
- Mantisa je ve formátu *jedna_číslice* + *řádová_čárka* + *1_až_n_číslic*
- Hodnota exponentu pak posouvá řádovou čárku
- Přesnost závisí na počtu bitů definujících mantisu, rozsah závisí na počtu bitů definujících exponent
- Výpočty:
    - U sčítání/odečítání se srovnají exponenty a sečtou/odečtou mantisy
    - U násobení se sečtou exponenty a vynásobí mantisy
    - U dělení se odečtou exponenty a vydělí mantisy
- Výhodou je maximální podpora ze strany HW i SW (`float`, `double`, ...)

Mikrořadiče

- Mikropočítač je realizace obvodů počítače na jedné desce plošných spojů (desktop, smartphone, PDA, server, superpočítač), obecný, důraz na HMI
- Embedded je mikropočítač integrovaný v jednom obvodu, důraz na úsporu, konkrétní činnost, machine-to-machine interface, program je obvykle vyvíjen na jiném systému
- Mikroprocesor je CPU realizovaný jedním obvodem s vysokou hustotou integrace; rozdělení:
    - Univerzální (CISC do desktopu, RISC do embedded)
    - Signálový (DSP – Digital Signal Processor) – filtry, FFT, modulace, ...
    - Grafické, multimediální, ...
- Mikrořadič je mikroprocesor doplněný pamětí a periferiemi (spíš RISC + Harvard (SRAM+FLASH) + střadač/registr ISA); obsahuje:
    - Čítače, časovače, watchdog timer
    - Porty, IO, sběrnice

    - Zdroj hodin
    - Přerušovací podsystémy
    - ADc, DAc, RTC

Paměti – zařízení pro uchování informací

- Dělení:
    - Neadresovatelné (FIFO/LIFO), adresovatelné (RAM/ROM), obsahem adresovatelné (asociativní – CAM)
    - Sekvenční/dynamický přístup
    - Optické, magnetodynamické, polovodičové, ...
    - Access time, cycle time (minimální doba mezi dvěma přístupy), kapacita, rychlost
    - Read and Write Memory:
        - DRAM (tranzistor+kapacitor, potřeba refresh, čtení je destruktivní) / SRAM (bistabilní klopný obvod, 6 tranzistorů)
    - Read Only Memory:
        - ROM / PROM / EPROM / EEPROM / FLASH
- (CPU) Cache:
    - Rychlá polovodičová paměť mezi rychlým a pomalým zařízením
    - L1 nejblíže ALU, stovky KB; L2 do 1 MB, každé jádro má svou; L3 jednotky MB, sdílená všemi jádry
    - Write-Through nebo Write-Back
    - Implementace na principu asociativní paměti, kde klíčem (tagem) je adresa (celá nebo její část); druhy:
        - Plně asociativní cache – celá adresa je tag a porovnává se s tagy v cache (mnoho komparátorů, velké tagy)
        - Přímo mapovaná (1cestná) cache – adresa se rozdělí na tag a *adresu třídy*; dekodér z *adresy třídy* vykouzlí číslo řádku, které se vybere z cache; pak se porovná tag vybraného řádku s hledaným tagem a máme cache hit nebo cache miss
        - n-cestně asociativní cache – máme n tabulek, každá se svými řádky; *adresa třídy* vybere řádek z každé tabulky (nejpoužívanější, má n komparátorů)
    - Parametry hit ratio / miss rate (jednoduché, ale hodně missuje)
    - Strategie mazání bloků při přeplnění: nejdéle nepoužívaný, nejméně používaný, nejstarší, náhodný
- Virtuální adresace – odkládání dat na disk, tváří se jako celistvý paměťový prostor
- DDRx DRAM – Double Data Rate – k přenosu dochází na náběžnou i sestupnou hranu hodin
- RAID – Redundant Array of Independent Disks – různé bloky na různé disky (rychlost) / mirroring / error detection & correction (velká redundance) / paritní disk

Sběrnice

- Informační cesta pro přenos informací mezi jednotlivými funkčními bloky systému, nebo pro propojení systému s okolím
- Soustava vodičů přenášejících data stejného charakteru, obvykle master/slave
- Dělení: dvou/vícebodové, dlouhé/krátké spoje, jedno/obousměrné, serial/parallel
- Sběrnice jsou dlouhé vedení, které je třeba impedančně zakončit terminátory
- Přenos: synchronní (synchronizační impulzy společně s daty nebo na vlastním vodiči; vhodné pro velké objemy dat; konstantní přenosová rychlost) / asynchronní (rámec obsahuje synchronizační informace;

vhodné pro nepravidelný přenos a odolnější proti rušení), arytmický (asynchronní domluva, dočasně synchronní přenos)

RS-232

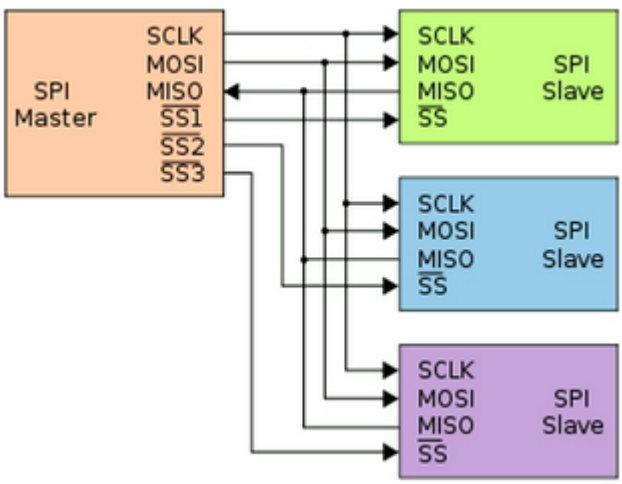
- Dříve velmi častý, dnes nahrazován USB; u počítače označován jako COM port (dnes virtuální COM port na sběrnici USB)
- Stále využíván v průmyslu (PLC, měřící přístroje, tiskárny, čtečky, ...)
- Sériová komunikace, stardard určuje pouze fyzickou vrstvu (napětí/časování/konektory/vlastnosti vedení) a neřeší přenášený protokol
- Dříve připojení počítače (DTE) k modemu (DCE); nebo lze zapojit dvě DTE
- Konektor CANON
- Rychlost 100 Kbitps, vzdálenost maximálně desítky metrů
- Přenos: `start bit` + `5–8 bitů` + `sudá parita / lichá parita / 0 / 1` + `jeden/dva stop bity`
- Signály RxD, TxD, ...

I^2C – Inter-Integrated Circuit

- Philips, platí se jim za přidělení unikátní adresy pro zařízení
- Multimaster sběrnice pro připojování (původně) pomalých periferií
  - Není pevně daný master; master je ten, kdo zrovna potřebuje vysílat
- Zapojení: EEPROM, porty, čidla, ADc/DAc, displeje, ..., standardní periferie
- Dva obousměrné vodiče; half-duplex synchronní sériový přenos
- Rychlost maximálně jednotky Mbitps a vzdálenost maximálně jednotky metrů
- Data se nikdy nesmí měnit, pokud jsou hodiny v jedničce, kromě start a stop podmínky
- `start` + `slave address` + (`8 bitů` + `acknowledge`)$^+$ + `stop`

SPI – Serial Peripheral Interface

- Motorola; (existuje podobná sběrnice Microwire)
- 4 vodiče, full duplex
- Každé další zařízení znamená vodič navíc ⇒ netřeba adresace, signál slave select
- Vždy jeden master – určuje hodiny a komunikační rychlost
- Všechny signály jsou jednosměrné – podporuje oba směry, ale ne na stejném drátu (MOSI+MISO – můžu najednou vysílat i přijímat)
- Použití pro stejné periferie jako u I^2C



USB – Universal Serial Bus

- Sériový přenos dat, 4/8 vodičů, vzdálenost jednotky metrů, až 127 zařízení, zpětná kompatibilita
- Podpora ovladačů v OS, autoidentifikace a překonfigurovatelné periferie, HotSwap
- Hierarchie: víceúrovňová hvězdicová (strom?), hub a node, pouze jeden hostitel

Paralelní porty

- Standard Parallel Port (Centronics) – ~8 datových vodičů stíněných GND vodiči + režijní vodiče; tiskárny
- Enhanced Parallel Port
- Extended Capabilities Port

Chipset

- Řídí komunikaci na základní desce, funguje jako řadič RAM i cache, může řešit power management
- Určuje typ/počet/kapacitu: procesorů, pamětí, rozšiřujících slotů, integrované prvky
- Obsahuje rozhraní pro PATA, SATA, USB, LAN, ...
- Definuje systémovou sběrnici mezi CPU a okolím
- Dříve severní a jižní můstek (komunikace přes DMI – Direct Media Interface)
- Dnes northbridge funkce v CPU a zbytek v *Controller Hub*, komunikace přes:
    - AMD HyperTransport, pak Infinity Fabric
    - Intel Quick Path Interconnect, pak Ultra Path Interconnect

PATA

- Kšandy, 2 zařízení na kabel („master/slave“ aby BIOS nebyl zmatenej)

SATA – Serial Advanced Technology Attachment

- Jeden kabel – jedno zařízení (point to point)
- 7 vodičů, délka až 1 metr
- Napájení není součástí rozhraní
- 600 MB/s

PCIe – Peripheral Component Interconnect Express

- Sériová point to point sběrnice složená ze dvou nízkonapěťových diferenciálních párů (každý pro jeden směr)
    - Tomuto páru se říká *lane* a sám by tvořil PCIe x1
    - Obvykle máme více *lanes*, které pracují nezávisle serial-ish-ově; proto PCIe nenazýváme paralelní sběrnicí
    - Až PCIe x32, víc *lanes* znamená také širší konektor
    - Základní rychlost je 250 MB/s, s každou verzí a s každým zdvojnásobením *lanes* se rychlost a propustnost zhruba zdvojnásobí

PCIe	**x1**	**x16**
v1.0	250 MB/s	4 GB/s
v6.0	~8 GB/s	~128 GB/s

M.2

- Náhrada za mini-SATA a mini-PCIe; zapojení SSD, Wi-Fi a Bluetooth modulů, ...
- Jedná se pouze o rozhraní, může na něm „běžet“ PCIe, NVME, SATA, ...

SAS – Serial Attached SCSI

- Skazina ¯_(ツ)_/¯
- Point to point, použití v serverech a diskových polích, HotSwap, stromová topologie, do které lze připojit i SATA disky

Speciální procesory

Signálové procesory

- Očekává se průběžné zpracování velkého množství dat „protékajících“ prcesorem v reálném čase
    - Na data se aplikují různé DSP algoritmy, má vysoký pracovní kmitočet a využívá co největšího stupně paralelizmu
- Konvoluce, korelace, filtrace (IIR, FIR), diskrétní transformace, práce s maticemi
    - Důraz na násobení konstantou/proměnnou, akumulaci dílčích součinů, aritmetické a logické posuny
- Dělení:
    - DPS mikroprocesory (neobsahují paměť ani další periferie) a DSP mikrokontroléry (periferie např. ADc/DAc)
    - Celočíselné (levné, složitější vývoj) a s plovoucí řádovou čárkou (složitější struktura, větší spotřeba, přesnější, jednodušší algoritmy)
- Obvyklá architektura:
    - Harvard, RISC, pipelining, hodně registrů (registrová ISA)
    - Superskalár/VLIW
    - MAC unit (multiply-accumulate), ALU, barrel-shifter

Grafické procesory (GPU)

- Vykreslování – 3D data na 2D obraz
- SIMD – jeden postup nad velkým množstvím dat; masivně paralelní architektura
- Shadery, řadič paměti, Texture Mapping Unit, Render Output Unit

NPU – Neural Processing Unit / TPU – Tensor Processing Unit

- AI, upravené GPU

Mobilní procesory

- ARM, důraz na nízkou spotřebu, „big.little“ (kombinace různých jader)

Spojení CPU + FPGA

- V FPGA lze vytvořit vhodné (i nestandardní) vstupně-výstupní periferie nebo provádět specifické výpočty
- Vyšší náklady, může být rychlejší a energeticky účinnější

Paralelní systémy

- Instruction: **SI** – v čase řešení problému běží jeden program, **MI** – více programů paralelně
- Data: **SD** – jeden zpracovávaný tok dat, **MD** – více toků dat
- **SISD** – „původní" Von Neumann
- **MISD** – několik programů zpracovává stejný datový tok (lze uvažovat vektorové počítače)
- **SIMD** – všechny jednotky provádí současně stejnou instrukci, ale na různých datech (GPU)
- **MIMD** – obecný typ paralelního systému

Superskalární architektura

- Typická u všeobecných CPU
- Závislost registrů a dat na instrukcích řeší sám CPU (schedule unit) → jednodušší programování
- Instrukce pro dílčí jednotky jsou jednoduché → jednodušší kompilátory, vyšší rychlost
- Kompatibilita mezi generacemi CPU
- Problematická optimalizace (programátor neví, co přesně se děje pod kapotou)

VLIW – Very Long Instruction Word

- Všechny výkonné jednotky jsou umístěny paralelně vedle sebe
  - Pokud nenastane kolize v použitých registrech, mohou pracovat nezávisle na sobě
  - Např. paralelně: dvě ALU, jedna násobička, jedna dělička, ...
- V jedné instrukci jsou uloženy operační kódy pro všechny jednotky (jinak by se žádný čas neušetřil)
  - Díky tomu není potřeba komplexního řadiče, instrukce mají ale velkou délku
  - Nevyužité pozice v instrukci je nutné vyplnit instrukcemi NOP
- Překladač se stará o kolize v registrech a skládá instrukce tak, aby bylo instrukční slovo co nejvíce zaplněno
- Závislé na konfiguraci procesoru – změna výkonných jednotek znamená změnu zápisu slova a celý program musí být znovu přeložen
  - Různé generace nejsou obecně zpětně kompatibilní
  - Není vhodné pro dynamickou kompilaci a interpretaci

Paralelní víceprocesorové systémy

- Cena a spotřeba CPU klesá a výkon (rychlost) relativně tolik nestoupá ⇒ paralelní systémy s více CPU
- Výkonnost většinou neroste lineárně s počtem přidaných CPU, je spíše logaritmická a od určitého bodu se nevyplatí přidávat další CPU
- Zdroje neefektivnosti: nedostatek / špatné rozdělení práce, velké komunikační náklady a režie synchronizace CPU
- MSIMD – několik nezávislých SIMD, SPMD – SIMD bez synchronizace
- Zálohované systémy – eliminace poruch
- Těsně vázané systémy – sdílená paměť (propojovací síť CPUs↔MEM), CPU mají při nejlepším velmi malou paměť, nároky na rozsah a rychlost komunikace mezi jednotlivými CPU
  - Po určitém bodě již nová CPU nezvyšují výkon → použití cache → problémy synchronizace
- Volně vázané systémy (multipočítače) – každý CPU vybaven velkou pamětí, slabá interakce mezi CPU, zasílání zpráv
  - Clustery – masivně paralelní počítače s rychlou propojovací sítí (výpočetní / dostupnost / rozložení zátěže)

Propojovací sítě

- Statické sítě – neměnné, méně složité sítě u volně vázaných systémů
    - Uzlově symetrické – kubická síť (1D/2D/3D/4D/...)
    - Uzlově nesymetrické – binární strom, hvězdice
    - Mřížka, válec, ...
- Dynamické sítě – obsahují spínací prvky, globální/lokální řízení
    - Křížový přepínač, sběrnice
- Ethernet < Myrinet < InfiniBand