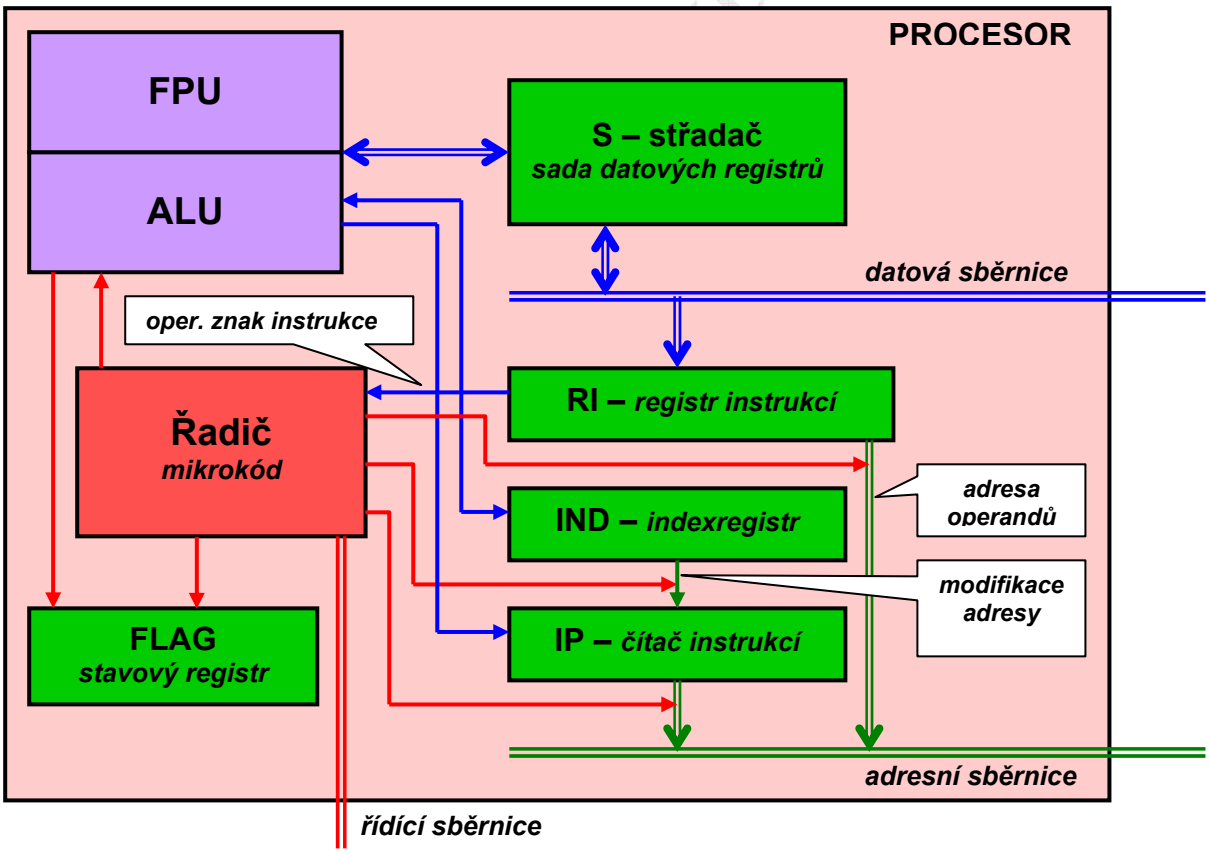
Procesor

Procesor

- Integrovaný obvod zajišťující funkce CPU
- Tvoří „srdce“ a „mozek“ celého počítače a do značné míry ovlivňuje výkon celého počítače (čím rychlejší procesor, tím rychlejší počítač)
- Provádí jednotlivé instrukce programu
- Synchronní zařízení, které pracuje podle hodinových kmitů generovaných krystalem umístěným na základní desce
- Většinou umístěn na základní desce

- **Procesor se skládá z těchto základních částí:**
  - *aritmeticko-logické jednotky (ALU)*
  - *jednotky pro práci s čísly v pohyblivé řádové čárce (FPU)*
  - *řadiče*
  - *registrů*
  - *případně vnitřní cache paměti (mezipaměti)*



Parametry procesoru

- **Frekvence (rychlost)**
  - počet strojových cyklů (taktů) nebo operací provedených za jednu sekundu
    - jednotka: **Hertz** [Hz] (např.: 4,77 MHz - 3,6 GHz)
    - jednotka: **MIPS** (milion instrukcí za sekundu)
  - současné základní desky (a zařízení k nim připojená) pracují s různými frekvencemi (odlišnými od frekvence procesoru)

- **Počet instrukčních kanálů (pipelines)**
  - udává maximální počet instrukcí proveditelných v jednom taktu procesoru
  - rozsah: 1 - 4 instrukční kanály

- **Šířka slova**
  - maximální počet bitů, které je možné zpracovat během jediné operace (např.: 8, 16, 32, 64 bitů)
  - určuje největší číslo, které procesor může zpracovat v rámci jedné operace
  - větší čísla musí být rozdělena na menší a zpracována po částech

- **Šířka přenosu dat**
  - maximální počet bitů, které je možné během jediné operace přenést z (do) čipu procesoru
  - je určena šířkou **datové sběrnice** procesoru
  - nezávisí na šířce slova
  - např.: 8, 16, 32, 64 bitů

- **Velikost adresovatelné paměti**
  - velikost paměti, kterou je procesor schopen adresovat (používat)
  - je dána šířkou **adresové sběrnice** a způsobem vytváření fyzické adresy
  - např.: 1 MB - 64 GB

Řadič

- **Řadič** načítá strojové instrukce, dekóduje je a řídí činnost procesoru při jejich provádění

- **Mikrooperace** – dílčí operace (např. zvýšení obsahu čítače, načtení operandu z hlavní paměti, generování řídících signálů pro ALU) nutná k provedení instrukce.
  - Pro tyto mikrooperace je vytvořen operační kód - **mikrokód**. Mikrokód se skládá z jednoduchých instrukcí a lze jej snadněji realizovat pomocí logických obvodů.

- **Mikroprogram** – program, který řídí činnost řadiče. Říkáme, že strojové instrukce jsou mikroprogramem interpretovány a řadič je realizován jako mikro programem řízený procesor.

CISC/RISC procesory

- **CISC** (*Complex Instruction Set Computer*)
  - **Mikroprogramový řadič** – obsahuje úplnou sadu strojových instrukcí včetně takových, které realizují poměrně složité operace.
  - Interpretace takto rozsáhlého strojového kódu mikroprogramem vedlo ke snížení výkonnosti procesorů.

- **RISC** (*Reduced Instruction Set Computer*)
  - **Hardwarově realizovaný řadič** – obsahuje pouze rychlé a jednoduché instrukce (omezená sada strojových instrukcí)

Řízení činnosti ALU

- **Činnost ALU** může být **řadičem řízena**

  - **Volně** - řadič pouze ALU zadá požadavek na vykonání operace a ALU pak tento požadavek plní samostatně. Takovéto ALU mají vlastní řadič a provádí samostatně i složitější aritmetické operace jako je násobení a dělení.

  - **Těsně** - ALU je jednoduchým kombinačním obvodem. ALU provádí jen základní aritmetické a logické operace. Násobení a dělení je realizováno buď mikroprogramem (CISC procesory) nebo je realizováno na úrovni strojového kódu systémovým programem (RISC procesory).

Ukázky procesorů

Zvyšování výkonu procesoru

Rozdělení strojové instrukce do fází

- **Výběr instrukce** (*fetch*). Instrukce je načtena z vyrovnávací paměti nebo z vnější paměti do vstupního bufru instrukcí.
- **Dekódování instrukce 1** (*decode stage 1*). Je dekódován operační kód instrukce a způsob adresace.
- **Dekódování instrukce 2** (*decode stage 2*). Jsou nastaveny řídící signály pro jednotku ALU. Pro složitější způsoby adresace je proveden výpočet adresy.
- **Provedení instrukce** (*execute*). ALU provede požadovanou operaci. Operandy čte z vyrovnávací paměti nebo z registrů. Pokud je operand v hlavní paměti, jsou vloženy prázdné cykly.
- **Uložení výsledku** (*write back*). Zápis výsledku zpět do registru nebo do paměti.

Klasické zpracování instrukcí

- Zpracování každé fáze trvá **jeden strojový cykl**
- Následující instrukce se začne provádět až **po skončení předcházející instrukce**
- Časový průběh zpracování strojových instrukcí i_1, i_2, … uvádí tabulka:

Strojový cykl	1	2	3	4	5	6	7	8	9	10	11
výběr instr.	i_1					i_2					i_3
dekódování1		i_1					i_2				
dekódování2			i_1					i_2			
provedení i.				i_1					i_2		
uložení výsl.					i_1					i_2	

Proudové zpracování instrukcí

- Proudové zpracování instrukcí zavádí omezený paralelizmus do procesu zpracování strojových instrukcí. Paralelizmu se dosahuje tím, že následující instrukce se začne provádět dříve, než zpracování předcházející instrukce skončí.
- Procesor navržen tak, že se **skládá z několika samostatně pracujících jednotek**. Každá jednotka zajistí část zpracování strojové instrukce. Jednotky, do kterých je procesor rozdělen, pracují paralelně a v daný okamžik zpracovávají různé strojové instrukce.

- Zpracování strojových instrukcí je rozděleno do **5 fází**.
- Zpracování každé fáze trvá **jeden strojový cykl**
- Časový průběh zpracování strojových instrukcí i_1, i_2, … i_8 uvádí tabulka:

Strojový cykl	1	2	3	4	5	6	7	8
výběr instr.	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
dekódování1		i_1	i_2	i_3	i_4	i_5	i_6	i_7
dekódování2			i_1	i_2	i_3	i_4	i_5	i_6
provedení i.				i_1	i_2	i_3	i_4	i_5
uložení výsl.					i_1	i_2	i_3	i_4

Nepodmíněný skok

- Procesor zpracovává instrukce i_1 , i_2 , po kterých následuje instrukce **nepodmíněného skoku i_3 na adresu 10**. Procesor ve strojovém cyklu 4 (i_3 ve fázi *decode* 1) rozpoznal, že se jedná o nepodmíněný skok. V cyklu 5 získal adresu, na kterou má být skok proveden, ukončil zpracovávání dosud rozpracovaných instrukcí, vynuloval vzniklé mezivýsledky a nastavil adresu nově načítané instrukce. Během strojového cyklu 6 načetl další instrukci (i_{10})

Strojový cykl	1	2	3	4	5	6	7	8	9	10
výběr instr.	i_1	i_2	i_3	i_4	i_5	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}
dekódování1		i_1	i_2	i_3	i_4		i_{10}	i_{11}	i_{12}	i_{13}
dekódování2			i_1	i_2	i_3			i_{10}	i_{11}	i_{12}
provedení i.				i_1	i_2	i_3			i_{10}	i_{11}
zápis výsled.					i_1	i_2				i_{10}

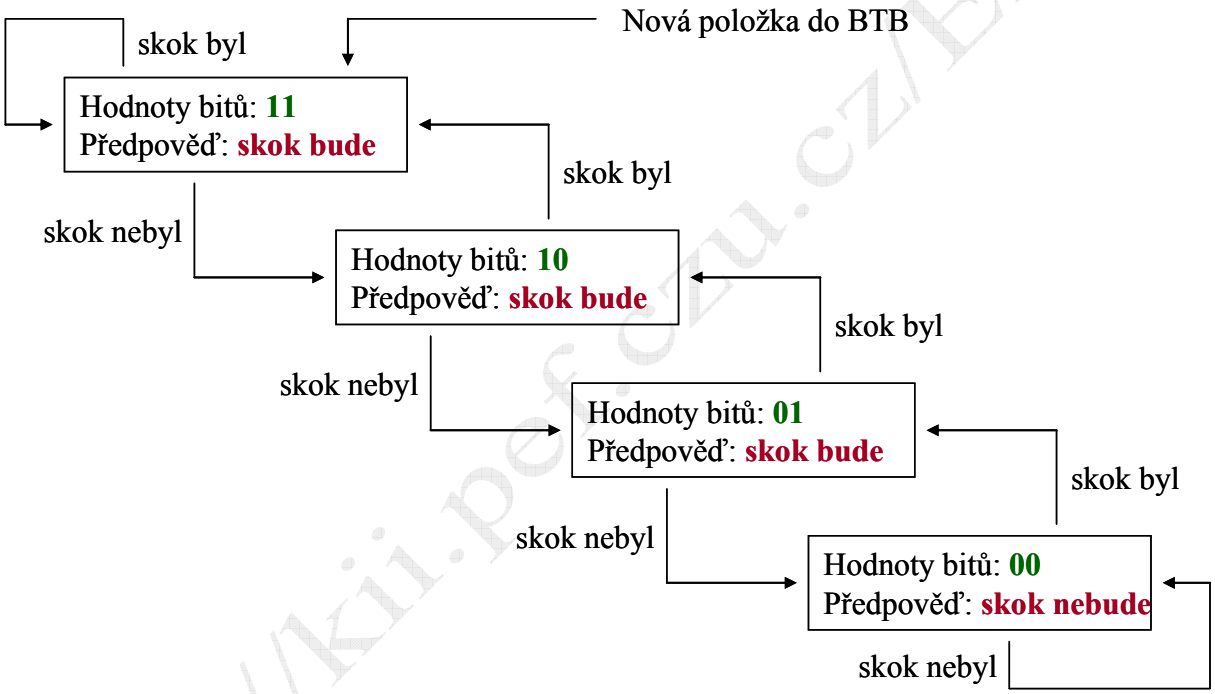
Podmíněný skok

- Pokud je ve vstupním proudu instrukcí instrukce **podmíněného skoku**, procesor zjistí, zda bude provádět skok nebo ne, až **ve fázi *provedení instrukce***.
- Ve fázi ***dekódování 1*** zjistí, že zpracovává instrukci podmíněného skoku:
  - pokud ve fázi ***provedení instrukce*** zjistí, že **skok nebude provádět**, pokračuje normálním způsobem – bez přerušení proudu instrukcí
  - pokud ve fázi ***provedení instrukce*** zjistí, že je třeba provést skok na danou adresu, resetuje registry a pokračuje instrukcí, na kterou má být skok proveden – proud instrukcí je na několik strojových cyklů přerušen.

- **Další narušení zpracování**
  - K narušení dojde i v případě, že obsah vstupního operandu instrukce je některou z blízkých předcházejících instrukcí změněn.

Predikce skoků

- Moderní procesory (např. Pentium) mají **jednotku pro predikci skoků**
- Tato jednotka si uchovává informaci o jednou již použitých skokových instrukcích a na základě těchto informací je schopna s předstihem (ve fázi ***dekódování***) predikovat, zda bude skok proveden a pokud ano, na jakou adresu. Na základě této predikce procesor načte a začne zpracovávat tu instrukci, která bude pravděpodobně následovat.
- Např. **paměť BTB** (*Branch Target Buffer*)



Superskalární a hyperskalární architektura

- U některých procesorů (např. Pentium) je **zdvojena ALU** a to umožňuje souběžné provedení dvou instrukcí – taková **architektura** se nazývá **superskalární**.
- Některé procesory (např. Pentium II, III) mají **tři ALU nebo dvě ALU s dvojnásobnou frekvencí** než je základní frekvence procesoru. To umožňuje provádět **3 až 4 instrukce** během jednoho strojového cyklu – taková **architektura** se nazývá **hyperskalární**.

- Časový průběh zpracování instrukcí u superskalární architektury uvádí tabulka:

Strojový cykl	1	2	3	4	5	6	7	8
výběr instr.	i_1 i_2	i_3 i_4	i_5 i_6	i_7 i_8	i_9 i_{10}			
dekódování1		i_1 i_2	i_3 i_4	i_5 i_6	i_7 i_8	i_9 i_{10}		
dekódování2			i_1 i_2	i_3 i_4	i_5 i_6	i_7 i_8	i_9 i_{10}	
provedení i.				i_1 i_2	i_3 i_4	i_5 i_6	i_7 i_8	i_9 i_{10}
uložení výsl.					i_1 i_2	i_3 i_4	i_5 i_6	i_7 i_8

Dynamické vykonávání instrukcí

- Tento způsob zpracování instrukcí byl poprvé zaveden u procesorů P6 (Intel) a je kombinací tří metod zpracování dat:

  - **předpovídání větvení** – procesor je schopen speciálním algoritmem předpovídat skoky nebo větvení v programu. Současně s načítáním instrukcí si procesor v paměti předběžně čte další instrukce

  - **analýza toku dat** – slouží k logické kontrole instrukcí a k jejich seřazení do optimálního pořadí bez ohledu na to, jak jsou zapsány v programu. Procesor nejprve zjistí, zda mohou být instrukce vykonány, či zda jsou závislé na dokončení jiných instrukcí

  - **spekulativní vykonávání instrukcí** – schopnost procesoru vykonávat instrukce, které se nacházejí až za aktuální polohou ukazatele v programu. Na základě analýzy toku dat se provedou ty instrukce, které nejsou závislé na dokončení jiných instrukcí a jejich výsledky jsou uloženy do dočasných registrů. I když jsou instrukce vykonány mimo pořadí, jejich výsledky jsou do paměti zapisovány v pořadí, odpovídajícím jejich zápisu v programu.

Technologie EPIC

- **Technologie EPIC** (*Explicitly Parallel Instruction Computing*)
  - všechny instrukce v **128 bitových balíčcích**
    - **balíček** = 3 x 41-bitové instrukce a 5-bitový informační kód
  - balíček načten najednou – podle informačního kódu zjištěn typ operace (celočíselná operace, operace s reálnými čísly, …)
  - jsou-li balíčky odlišného typu je možné zpracovat najednou
  - libovolný počet balíčků lze seskupit do **superbalíčku** – instrukce se vzájemně neovlivňují → mohou být prováděny paralelně v libovolném pořadí
  - až **20 instrukcí** během jednoho cyklu