Přerušení a výjimky

Milan Radojčić

SSPŠaG – Operační systémy

26. 3. 2026

Opakování: Úvodní kvíz

Operační systém[1] běží:

A) od začátku spuštění počítače, aktivně monitoruje procesy.
B) kdykoliv o to některé z vláken požádá.
C) při spuštění počítače a poté jen někdy, reaktivně.
D) vždy na pozadí na jednom z jader. Monitoruje chování procesů, aby mohl reagovat správným způsobem[2].
E) jen při vzniku a zániku vláken.

[1] Myslí se kernel.

[2] Např. ukončit proces, když provede něco nekalého.

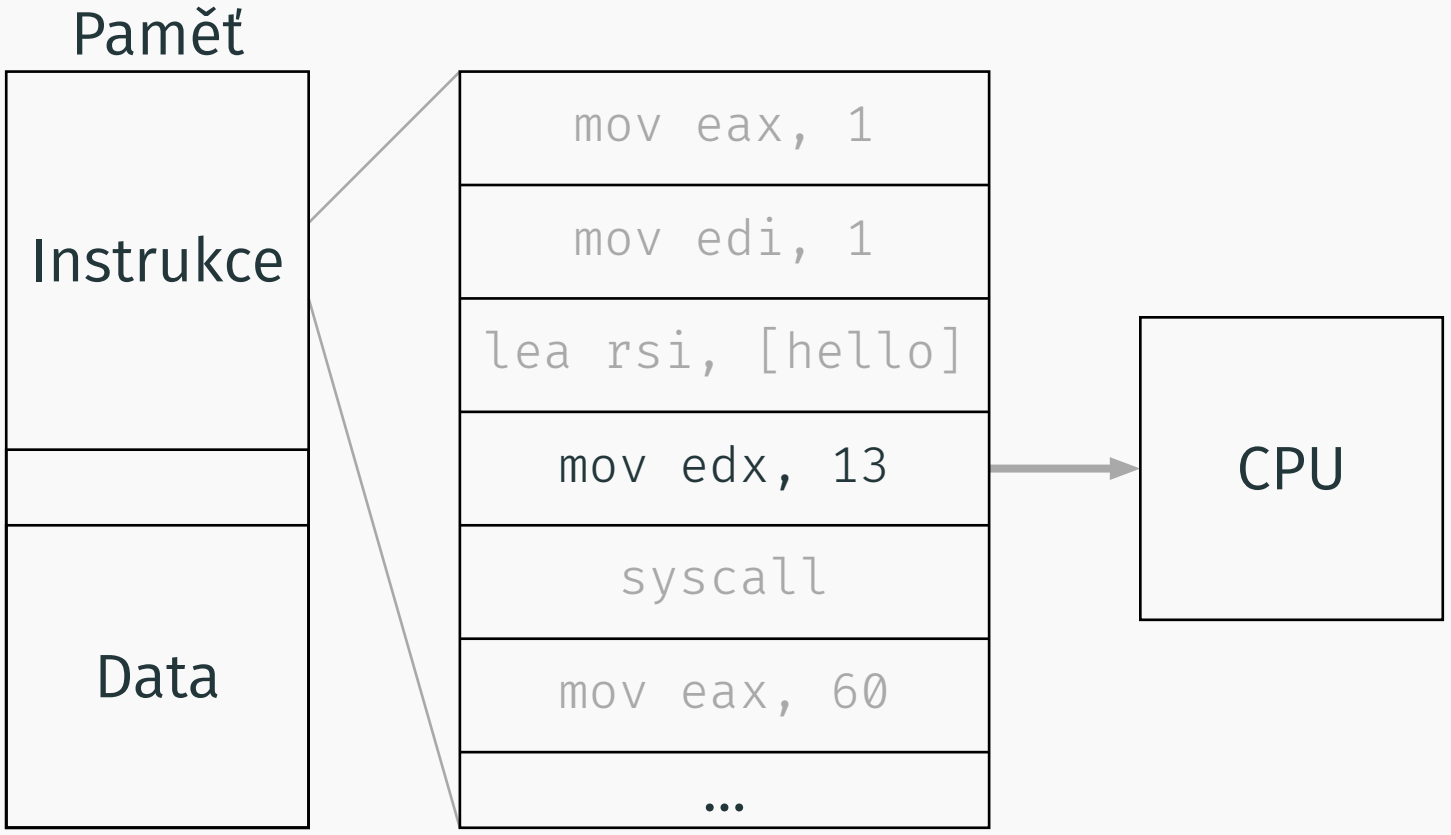
Opakování: Úvodní kvíz - správné odpovědi

Operační systém běží:

A) od začátku spuštění počítače, aktivně monitoruje procesy.
B) **kdykoliv o to některé z vláken požádá.**
C) **při spuštění počítače a poté jen někdy, reaktivně.**
D) vždy na pozadí na jednom z jader. Monitoruje chování procesů, aby mohl reagovat správným způsobem.
E) jen při vzniku a zániku vláken.

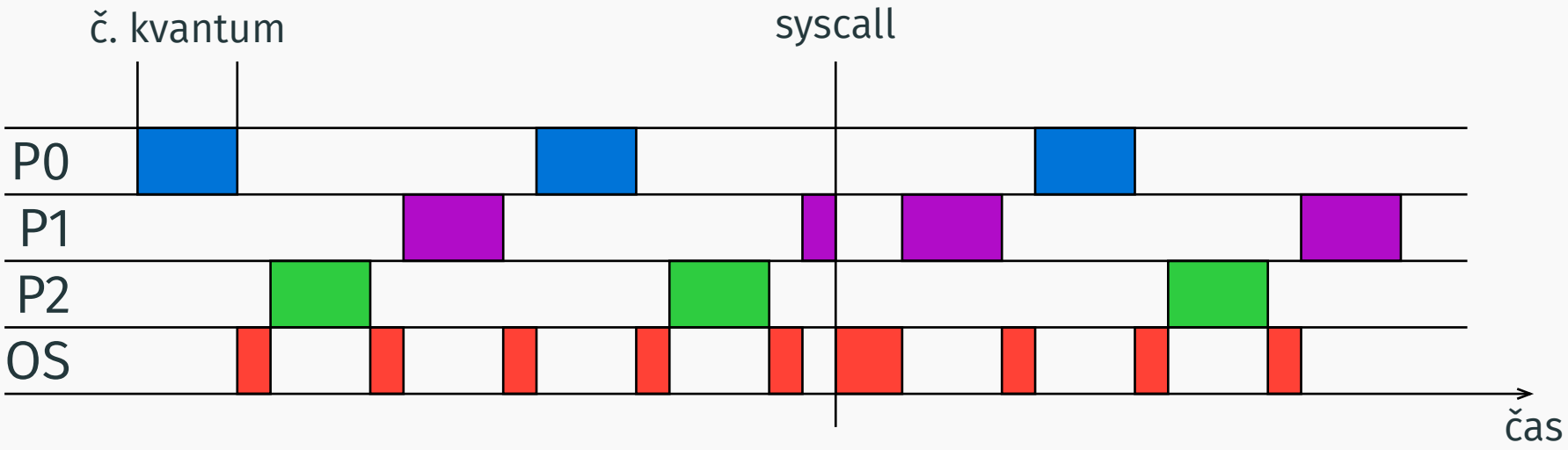
Opakování: Co dělá procesor?

- Procesor jednoduše **načítá** instrukce a **spouští** je jednu za druhou.



- Jeden procesor spouští **pouze jeden proud instrukcí**.

Opakování: Preemptivní plánování



Při **preemptivním přístupu** je vláknům přiřazeno nějaké **časové kvantum**. Po něm je jim procesor odebrán. Vlákna se mohou vzdát procesoru před vypršením kvanta pomocí systémového volání.

Jak ví hardware, že má přestat spouštět proces a začít spouštět kód OS?

Přerušení a výjimky

Přerušení

Přerušení jsou události, které mění posloupnost vykonávaných instrukcí.

- Při přerušení se přestane spouštět aktuální kód a **začne se spouštět kód na daném místě v kernelu**.

Vnější vs. vnitřní přerušení

Přerušení dělíme na vnitřní a vnější:

- **Vnitřní** (synchronní, výjimky) – Přichází ze samotného CPU. Často se jedná o chyby (např. dělení 0), někdy můžou být validní (page fault).
- **Vnější** (asynchronní) – Přichází z I/O zařízení: stisky kláves, komunikace s HDD, atd.

Přerušení a výjimky

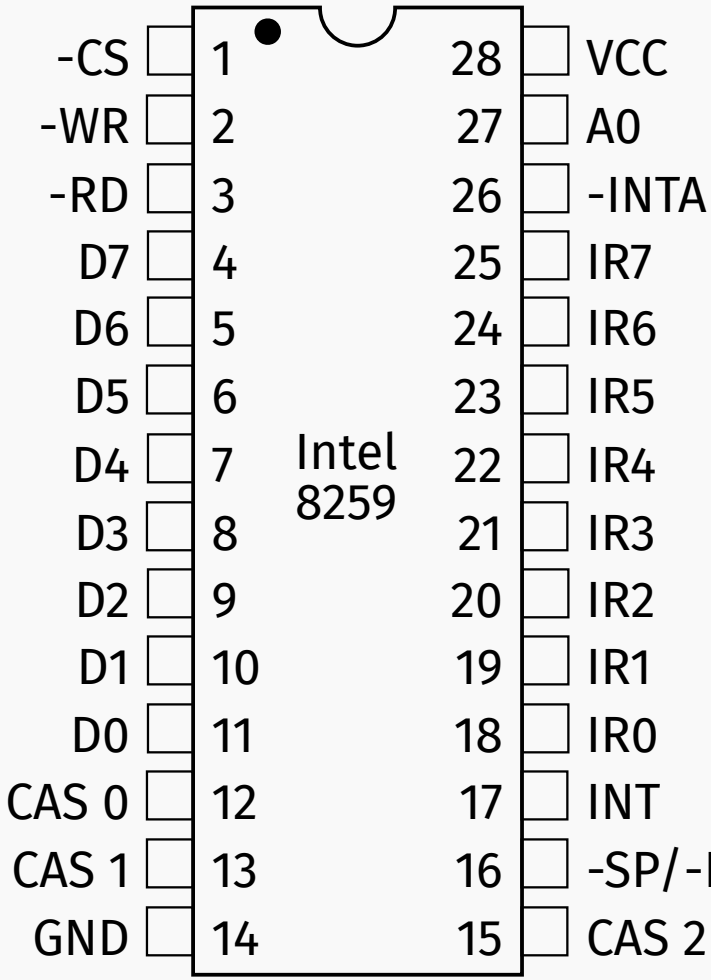
Ke každému přerušení patří **kód** (angl. *interrupt vector*). Tohle mapování lze měnit.

Maskovatelné a nemaskovatelné přerušení

- Některé přerušení lze na chvíli ignorovat, těm říkáme **maskovatelné** (např. timery, klávesnice, atd.).
- Jiné nelze ignorovat, těm říkáme **nemaskovatelné**. Jsou to hlavně závažné HW chyby.

PIC Intel 8259

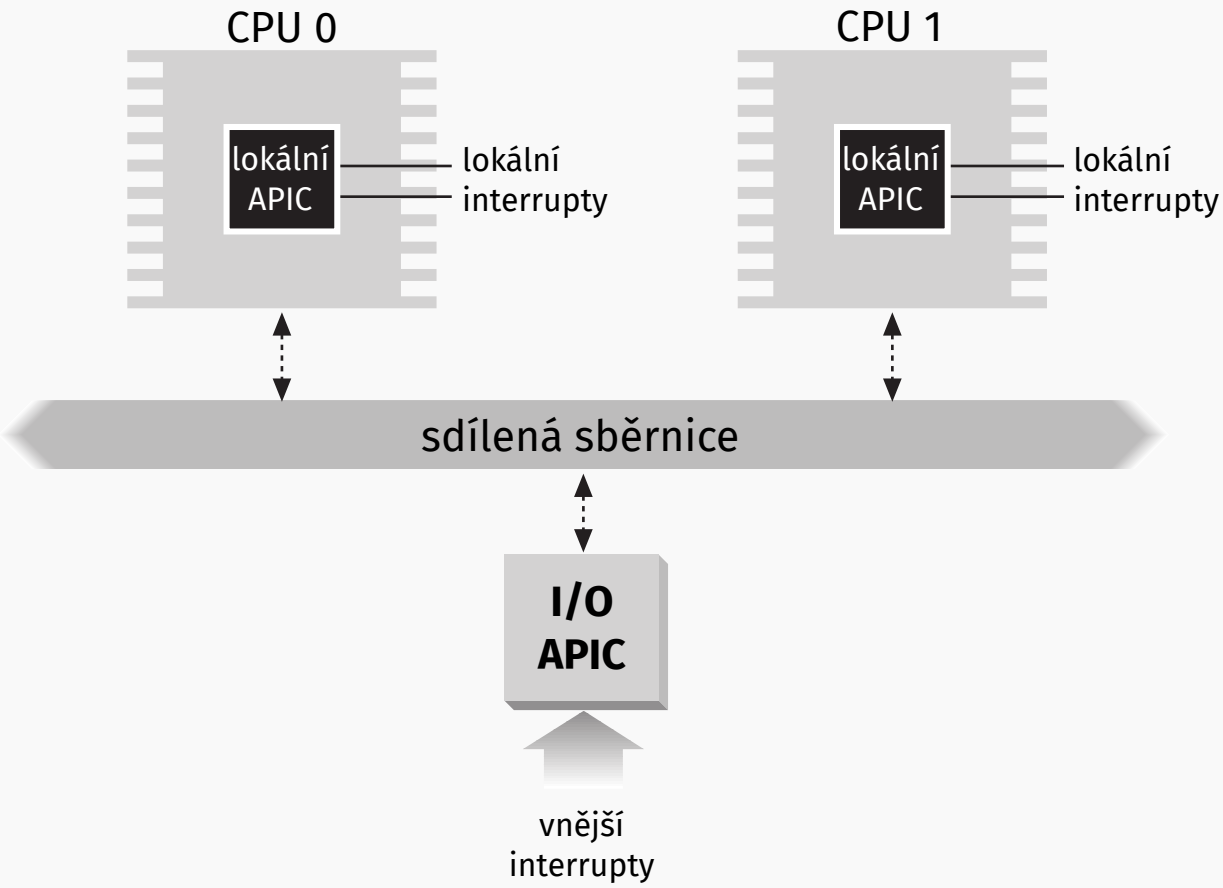
- PIC je zkratka pro **Programmable Interrupt Controller**.
  - Jednotlivé interrupty liny jdou vypnout.
  - Jde upravovat mapování interrupt-kód.
- Prakticky funguje jako multiplexer: několik IR0-IR7 pinů dává dohromady do INT pinu pro CPU.



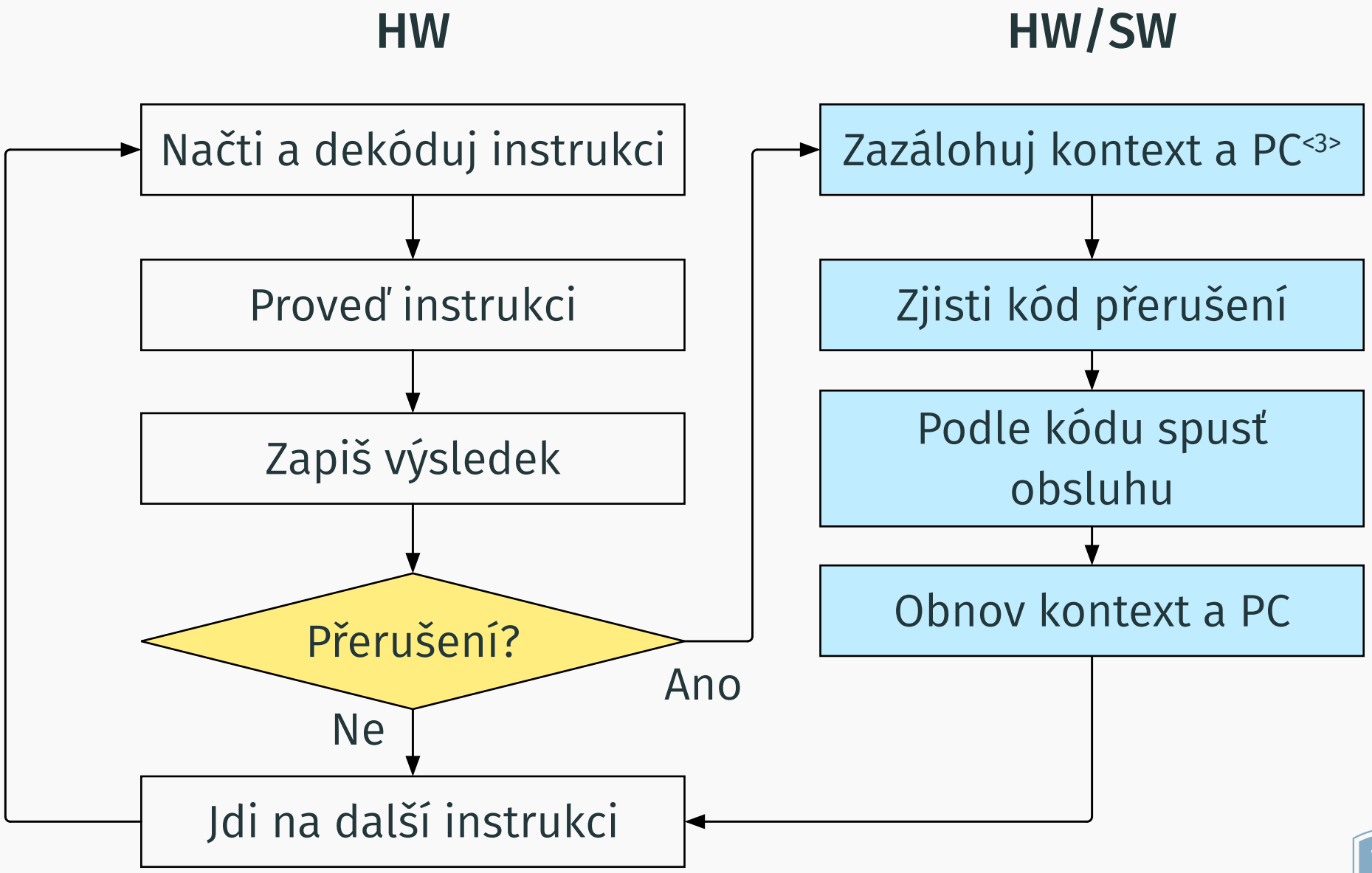
Pinout Intel 8259

Advanced PIC (APIC) architektura

- Situace se komplikuje, když máme více CPU.
- Vnější přerušení chceme dostat **k jen 1 CPU!**
  - Rozmyslete proč.
- Intelovská APIC architektura umí rozřazovat přerušení např na základě priority.



Obsluha přerušení



<3> PC = program counter – ukazatel na aktuálně zpracovávanou instrukci

Obsluha přerušení

- K obsloužení přerušení musí HW vědět **kde se v paměti nachází kód obsluhující rutiny**.
- Po spuštění nahraje kernel do CPU ukazatel na tabulku, která toto obsahuje.
- Na architektuře x86 je 256 možných kódů přerušení.

Offset	Obsah
0	Adresa obsluhy pro kód 0
1	Adresa obsluhy pro kód 1
2	Adresa obsluhy pro kód 2
...	...
255	Adresa obsluhy pro kód 255

Zjednodušená tabulka obsluh přerušení

Opakování: Systémová volání

- Jeden z hlavních způsobů jak interagujeme s OS jsou **systémová volání**.
- Je to vlastně požadavek pro OS, aby něco za nás udělal, např:
  - Spustil proces `fork`, `execve`; ukončil proces `kill`.
  - Otevřel soubor `open`, zapsal do něj data `write`.
- Systémová volání připomínají volání knihovních funkcí.
- Můžeme si o nich přečíst víc pomocí `man`.

Jak se tedy systémová volání liší od knihovních?

Systémová volání a přerušení

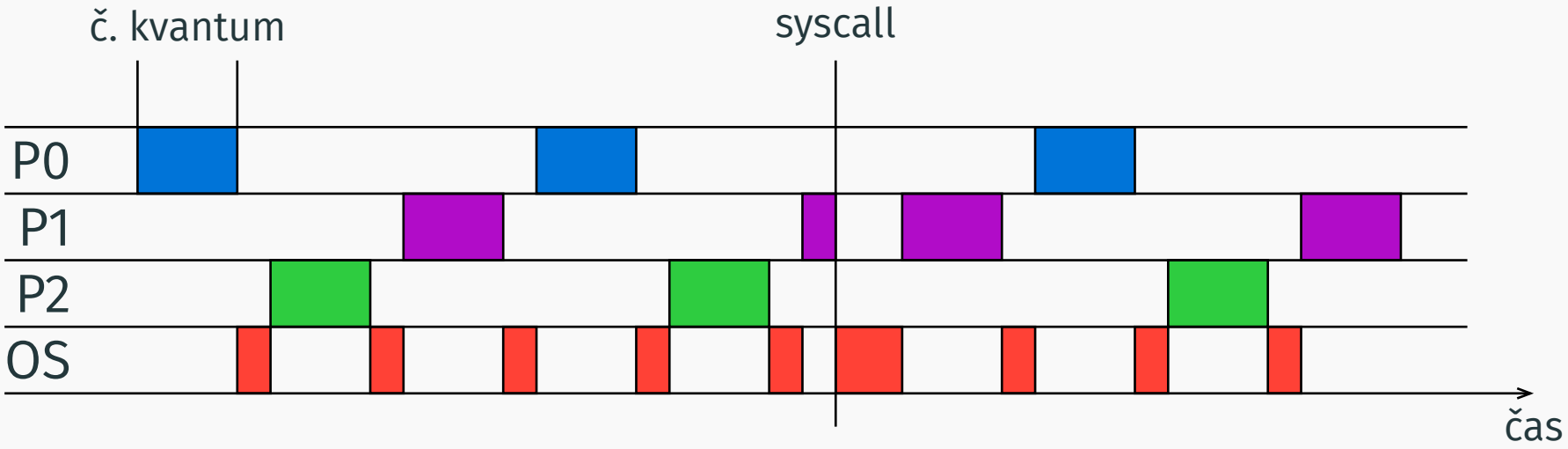
- V Linuxu odpovídá systémovým voláním kód přerušení 128 (`0×80`).
- Systémové volání lze provést pomocí instrukcí `int 0×80` nebo `syscall`.

Průběh systémového volání

OS	HW	Proces
Boot		
Inicializace tabulky přerušení		
Načtení programu do paměti, atd.		
...	...	...
		`main()`
		syscall
	Uložení kontextu	
Proběhne obsluha přerušení		
	Obnova kontextu	
		Spouští se dál po instrukci `syscall`

Preemptivní plánování a přerušení

- Preemptivní plánování také využívá přerušení.
- Některé přerušení vyvolávají **časovače** – ty lze využít k periodickému probouzení kernelu.



Průběh přeplánování

OS	HW	P0	P1
		Běží	
	Přerušení od časovače		
	Uložení kontextu		
Uložení kontextu P0 do struktur kernelu[4]			
Vybere se další proces ke spuštění			
Nahraje se kontext P1 ze struktur kernelu			
	Obnova kontextu		
			Běží

[4] Důvodem je to, že těsně po přerušení bydlí kontext P0 na stacku kernelu. My ho chceme někam zazálohovat permanentněji.

Stránkování a page fault

- Linux dělá hodně věcí líně.
- Např. když chce program používat nějakou knihovnu, tak není ihned nahrána do paměti.
- Při prvním přístupu do paměti knihovny není v paměti validní překlad stránky.
  - Nastane vnitřní přerušení **page fault**.
- Linux detekuje, že příslušná stránka není ještě načtená v paměti.
  - Načte ji, upraví tabulku stránek a **znovu spustí instrukci, která vyvolala page fault**.

Stránkování a page fault

- Page fault může nastat v případě, kdy se proces chová divně.
  - Je teda nutné rozeznávat validní a nevalidní page fault.
- Proces může kvůli nějaké chybě přistoupit k paměti, která vůbec není inicializovaná.
- V takovém případě zase nastane **page fault**.
- OS detekuje, že tato paměť není inicializovaná a je to tedy opravdový page fault.
- Procesu je poslán signál `SIGSEGV`[5] a většinou je ukončen.

[5] segmentation violation

TLB a meziprocesorové přerušení

Co může být problém při odkládání stránky na disk, když máme víc procesorů?

- Ostatní procesory můžou mít uložený překlad v TLB.
  - Nevěděli by, že je stránka odložená na disk a sahali by do náhodné paměti.
- K tomuhle existuje *meziprocesorové přerušení*, které **přinutí ostatní CPU, aby vymazali TLB.**

Shrnutí

- Popsali jsme, co jsou přerušení a jak je CPU zpracovává.
- Ukázali jsme několik příkladů přerušení, které jsou zajímavé pro OS.
- Hlavní zodpovědností OS je obsluha těchto přerušení.
- Zajímavý článek o historii přerušení pro zájemce: https://people.computing.clemson.edu/~mark/interrupts.html

Cvičení

- Pomocí `vmstat -w 1` můžete sledovat počet interruptů za vteřinu (sloupeček `in`).
- Jaké věci ovlivňují počet interruptů? Kdy jich je víc a kdy míň?