

# OPERAČNÍ SYSTÉMY

## První rozšířené zadání

**H**olubyev

**K**ruták

**M**ašíček

# Proč toto téma?

- Obsahově nám bylo nejsympatnější
- Bylo možno na něm pracovat už od napsání prvního základního zadání

# Více procesorů...

- **Boot...**
  - 0tý CPU – inicializace všech struktur
  - poté ostatní CPU vytvoří vlastní idle vlákno a čekají na reschedule event
  - každé CPU má vlastní safe stack...
- **Inter-CPU komunikace**
  - doručování zpráv pomocí interruptů (dorder)  
+ per CPU fronta zpráv
- **Zamykání všeho (malloc, threads, timer...)**
  - `disable_interrupts` nestačí => `BIG_LOCK`:  
`DisableInt` + `SpinLock` & `SpinUnlock` + `EnableInt`

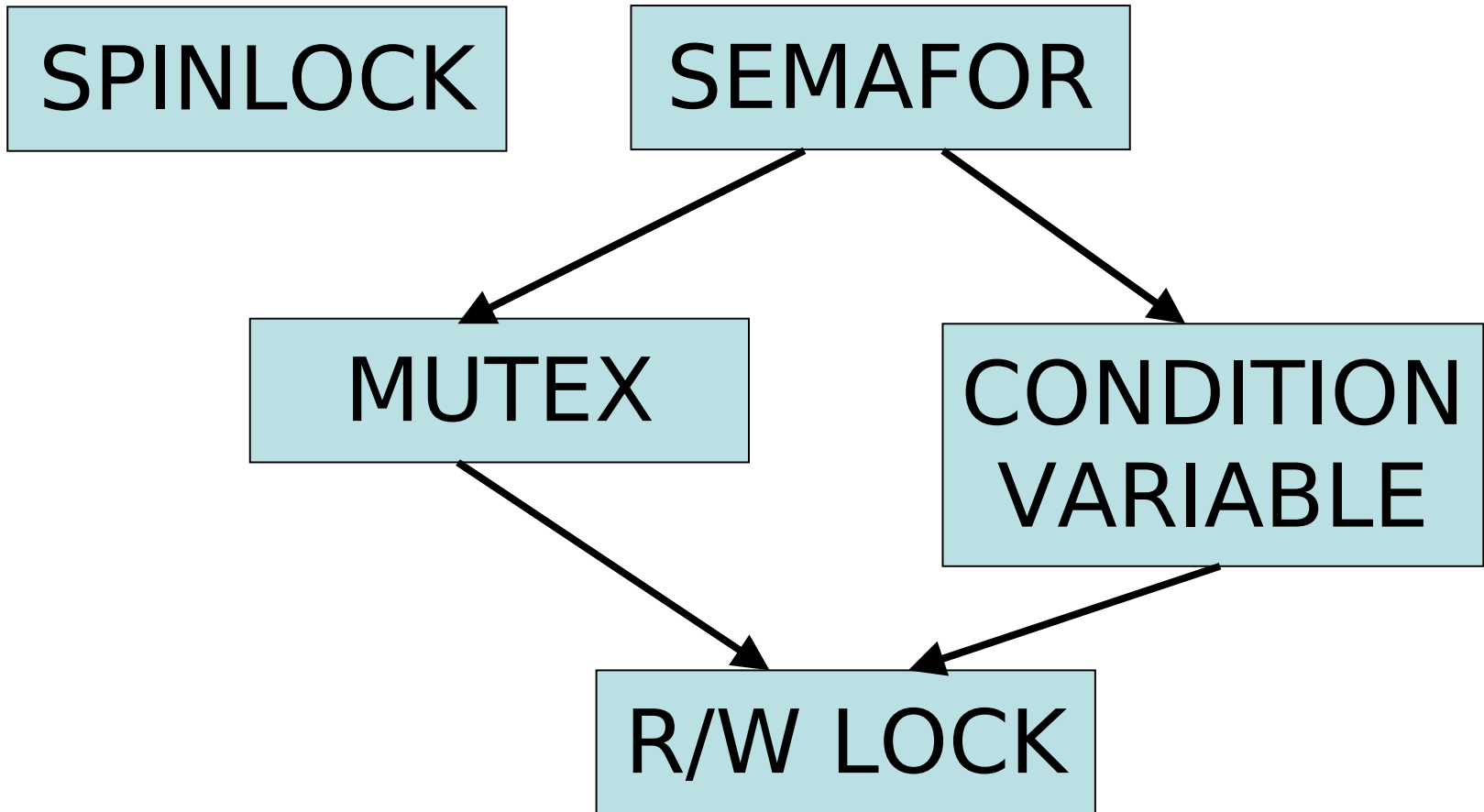
# Timer, Scheduler

- **Timer – distribuovaný**
  - na každém CPU běží vlastní count/compare i obsluha přerušení...
- **Plánovač - „distribuovaný v rámci mezí“ :-)**
  - každý CPU smí přeplánovat na další vlákno své fronty
  - jen 0tý CPU spouští časovač plánovače a vyvažuje fronty
    - je nutno se vyvarovat přesunu vlákna mezi CPU, pokud právě na nějakém běží
    - „u nás“ ale může být spuštěna obsluha časovače scheduleru na kterémkoliv CPU => přesměrování „zprávy“ na 0tý CPU

# Synchronizační primitiva

- **Jen jedno pracující se strukturou vlákna**
  - „linked listy“ vláken, uspávání vlákna, ...
- **Ostatní odvozená od něj**
  - (+) lepší rozšiřitelnost (např. na více procesorech)
  - (+) systematyczťější, přehlednější
  - (-) prvotní primitivum je složitější
  - (?) efektivita (optimalizátor C)
- **Vyjímkou je SpinLock**

# Návaznost synch. primitiv



# Neblokující seznamy

- **Bez použití zámků**
  - na principu compare and swap
- **Jednosměrný s ukazateli Head a Tail**
- **Přidávání prvků**
  - append, insert\_sorted, insert\_first
- **Mazání prvků**
  - delete
- **Vyhledávání prvků**
  - find

# Dotazy

**Děkujeme za  
pozornost**

**Dmytro Holubyev**

**Andrej Kruták**

**Viktor Mašíček**

**[www.cabadaj.net/hokrma\\_prezentace.pdf](http://www.cabadaj.net/hokrma_prezentace.pdf)**