HOGESCHOOL ROTTERDAM

Knowledge Clip

Embedded Systems

pthread semaphore

brojz@hr.nl

Mutex

- Simple way to create a **mutual exclusive** so called critical section.
  - Only **one** task can be in the critical section.

- Mutex has a **lock** (take) and a **unlock** (give) function.
  - OS ensures that these functions are **atomic**!
  - At the start of the critical section the mutex must be locked (taken) and at the end of the critical section the mutex must be unlocked (given).

Manage resources when more resources are available

Four men can enter, fifth one must wait until someone leaves.

Counting Semaphore

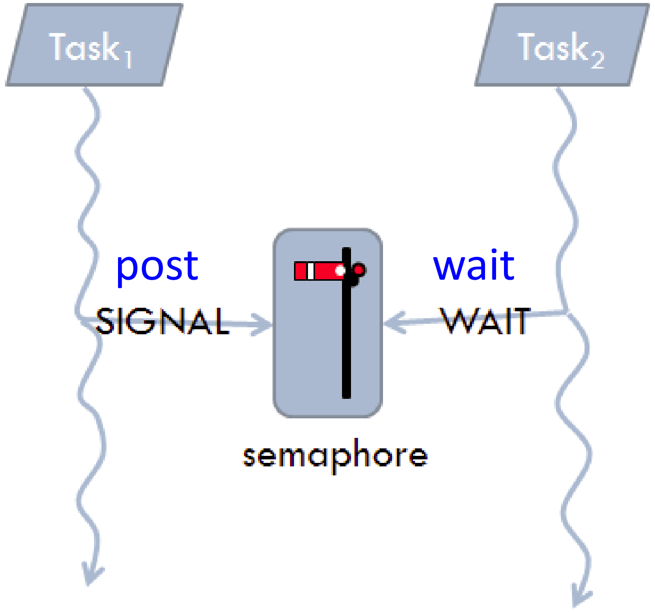
- Operations:
  - Psem (prolaag (probeer te verlagen), take, wait):
    wait (block, sleep) if `count == 0` else decrement `count`.
  - Vsem (verhoog, signal, give, post):
    unblock a waiting task if `count == 0` else increment count.

- Order of unblocking (wake up):
  - general purpose: FIFO
  - real-time: highest priority

Edsger Dijkstra

Semaphore versus Mutex

- Mutex can only be used for mutual exclusion (task which takes the mutex should also give the mutex (back)).
- Semaphore can also be used for other synchronization purposes.


Task$_1$
Task$_2$
post
SIGNAL
wait
WAIT
semaphore