# POSIX semaphores and mutexes

Paolo Burgio paolo.burgio@unimore.it





## Semaphores

- > A semaphore is a counter managed with a set of primitives
- > It is used for
  - Synchronization
  - Mutual exclusion
- > POSIX Semaphores can be
  - Unnamed (local to a process)
  - Named (shared between processed through a file descriptor)



## **Unnamed semaphores**

- > Mainly used with multithread applications
- > Operations permitted:
  - initialization /destruction
  - blocking wait / nonblocking wait
    - > counter decrement
  - post
    - > counter increment
  - counter reading
    - > simply returns the counter



## Initializing a semaphore

> The sem t type contains all the semaphore data structures

```
int sem_init(sem_t *sem, int pshared,
  unsigned int value);
  - pshared is 0 if sem is not shared between processes

int sem_destroy(sem_t *sem)
  - It destroys the sem semaphore
```



## **Semaphore waits**

```
int sem_wait(sem_t *sem);
int sem_trywait(sem_t *sem);
```

- > Under the hood..
- > If the counter is greater than 0 the thread does not block
  - sem\_trywait never blocks
- > sem wait is a cancellation point



## Other semaphore primitives

```
int sem_post(sem_t *sem);

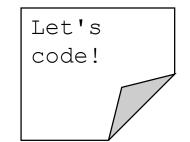
- It increments the semaphore counter
- It unblocks a waiting thread

int sem_getvalue(sem_t *sem,int *val);
```

It simply returns the semaphore counter



## **Example**



- > Filename: ex sem.c
- > In this example, semaphores are used to implement mutual exclusion in the output of a character in the console.



#### What is a POSIX mutex?

- > Like a binary semaphore used for mutual exclusion
  - But.. a mutex can be unlocked only by the thread that locked it

- > Mutexes also support some RT protocols
  - Priority inheritance
  - Priority ceiling
  - They are not implemented under a lot of UNIX OS

> Out of scope for this course



#### **Mutex attributes**

> Mutex attributes are used to initialize a mutex

```
int pthread_mutexattr_init (pthread_mutexattr_t *attr);
int pthread_mutexattr_destroy (pthread_mutexattr_t
  *attr);
```

> Initialization and destruction of a mutex attribute



### **Mutex initialization**

> Initialize a mutex with a given mutex attribute

> Destroys a mutex

```
int pthread_mutex_destroy (pthread_mutex_t *mutex);
```



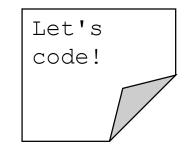
### Mutex lock and unlock

- > This primitives implement the blocking lock, the non-blocking lock and the unlock of a mutex
- > The mutex lock is **NOT** a cancellation point

```
int pthread_mutex_lock(pthread_mutex_t *m);
int pthread_mutex_trylock(pthread_mutex_t *m);
int pthread_mutex_unlock(pthread_mutex_t *m);
```



## Example



- > Filename: ex mutex.c
- > This is prev. example written using mutexes instead of semaphores.



### How to run the examples



> Download the Code/ folder from the course website

> Compile

\$ gcc code.c -o code -lpthread

- > Run (Unix/Linux)
- \$ ./code
- > Run (Win/Cygwin)
- \$ ./code.exe



#### **Useful links**



- > Course webpage
  - https://hipert.unimore.it/people/paolob/pub/Calcolo Parallelo/
- > Course GitHub
  - https://github.com/HiPeRT/cp19/



- > My contacts
  - paolo.burgio@unimore.it
  - http://hipert.mat.unimore.it/people/paolob/
- > PThreads
  - https://computing.llnl.gov/tutorials/pthreads/
  - http://man7.org/linux/man-pages/man7/pthreads.7.html
- A "small blog"
  - <a href="http://www.google.com">http://www.google.com</a>