OS Tutorial 4: Thread

Huan Wang
huanwang@uvic.ca

Outline

- **Pthreads API**
  - Thread Creation, Attributes & Termination
  - Sample Codes
- **Thread Synchronization**
  - Mutual Exclusions (Mutex)
  - Condition Variables (Convar)
- **Multithread Pitfalls & Debugging**

Outline

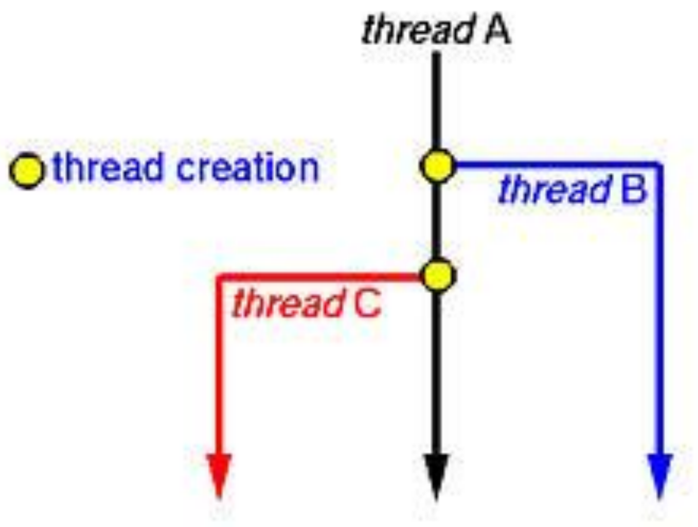
- **Pthreads API**
  - Thread Creation, Attributes & Termination
  - Sample Codes
- **Thread Synchronization**
  - Mutual Exclusions (Mutex)
  - Condition Variables (Convar)
- **Multithread Pitfalls & Debugging**

Thread Creation (1)

- Header file:
  - #include <pthread.h>
- Function call:
  - int **pthread_create**(pthread_t *thread, pthread_attr_t *attr, void *(*start_routine)(void *), void *arg);
- Arguments:
  - **thread**: returns the thread ID (pthread_t: an unsigned long int)
  - **attr**: attribute object of a thread (can be NULL)
  - **start_rounte**: a function to be executed by the created thread
  - **arg**: arguments for the function (can be NULL)
- Return values:
  - On success: return zero
  - On error: return **errno** (an nonzero **int** variable declared in <errno.h>)

Thread Creation (2)

- Once created, threads are peers and can create other threads.
- No implied threads hierarchy.
- No dependency between threads (except for main thread).



Thread Attributes (1)

- By default, a thread is created with certain attributes.
- Or various thread attributes can be assigned at the time of creation via thread attribute object (the second argument).
- initialize / destroy a thread attribute object:
  - **int pthread_attr_init(pthread_attr_t *attr);**
  - **int pthread_attr_distroy(pthread_attr_t *attr);**
- Other routines to query or set specific attributes of the object:
  - **int pthread_attr_setdetachstate(pthread_attr_t *attr, int detachstate);**
  - **int pthread_attr_setstackaddr(pthread_attr_t *attr, void *stackaddr);**
  - **int pthread_attr_setstacksize(pthread_attr_t *attr, size_t stacksize);**
  - …

Thread Attributes (2)

Thread attributes :

```
typedef struct {
    int __detachstate;
    int __schedpolicy;
    struct sched_param __schedparam;
    int __inheritsched;
    int __scope;
    size_t __guardsize;
    int __stackaddr_set;
    void *__stackaddr;
    unsigned long __stacksize;
} pthread_attr_t;
```

Thread Termination (1)

* **pthread_exit()** – *when a thread terminates itself.*
  * Header file: #include <pthread.h>
  * Function Call: void pthread_exit(void *retval);
  * Argument: **retval** – return value of pthread_exit().
* The returned value specifies an optional termination status that is typically returned to another thread calling **pthread_join()**.

Thread Termination (2)

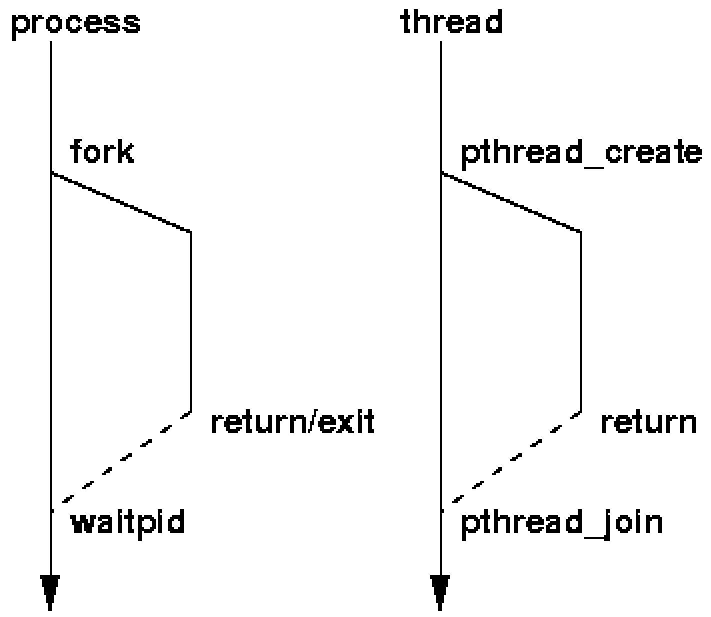
- **pthread_cancel()** – *when a thread wants to terminate another thread.*
  - Header file: #include <pthread.h>
  - Function call: int pthread_cancel(pthread_t thread);
  - Argument: **thread** – the ID of thread that will be terminated
  - Return values:
    - On success: return 0
    - On error: return **errno**

Thread Termination (3)

- **pthread_join()** – *when a thread waits for the termination of another thread.*
  - Header file: #include <pthread.h>
  - Function Call: int pthread_join(pthread_t thread, void **retval);
  - Argument:
    - **thread** – the ID of a thread that the current thread are waiting for.
    - **retval** – returned value from the terminated thread (can be **NULL** or the **retval** from **pthread_exit()**).
  - It suspends the calling thread until termination of the specified thread:
    - On success: return 0
    - On error: return **errno**

Thread Termination (3)

- **pthread_join()** – *when a thread waits for the termination of another thread.*


process
fork
return/exit
waitpid
thread
pthread_create
return
pthread_join