Semaphores, Condition Variables, and Mutexes in C

44-550: Operating Systems

Semaphores

- Defined in `semaphore.h`
- Semaphore type: `sem_t`
- Useful functions:

  -
    ```
    int sem_init(sem_t * sem, int pshared,
            unsigned int value)
    ```
  - `int sem_destroy(sem_t * sem);`
  - `int sem_wait(sem_t * sem);`
  - `int sem_post(sem_t * sem);`

Condition Variables

- Are always used in conjunction with a mutex
- Condition Variable Type: `pthread_cond_t`
- Useful functions:

  -
    ```
    int pthread_cond_init(pthread_cond_t * cond,
            const pthread_condattr_t * attr);
    ```

  -
    ```
    int pthread_cond_destroy(pthread_cond_t * cond);
    ```

  -
    ```
    int pthread_cond_signal(pthread_cond_t * cond);
    ```
    - Waits for the condition variable to be true, and locks the mutex

  -
    ```
    int pthread_cond_broadcast(pthread_cond_t * cond);
    ```

  -
    ```
    int pthread_cond_wait(pthread_cond_t * cond,
            pthread_mutex_t * mutex)
    ```

Mutexes

- Using a mutex requires four distinct steps:
  1. Creation/Initialization
  2. Locking
  3. Unlocking
  4. Destruction
- Because C does not have the concept of classes (with constructors and destructors), we must manually perform initialization and destruction.

Mutex Functions

Four functions for four steps:

-
  ```
  int pthread_mutex_init(pthread_mutex_t * mut,
          const pthread_mutexattr_t * attr);
  ```

-
  ```
  int pthread_mutex_lock(pthread_mutex_t * mut);
  ```

-
  ```
  int pthread_mutex_unlock(pthread_mutex_t * mut);
  ```

-
  ```
  int pthread_mutex_destroy(pthread mutex_t * mut,
          const pthread_mutexattr_t * attr);
  ```

Initializing Mutexes

pthread_mutex_init

```
int pthread_mutex_init(pthread_mutex_t * mut,
        const pthread_mutexattr_t * attr);
```

- Intitializes the mutex with the specified attributes
- `attr` may be NULL, uses default attributes
- Returns 0 on success, error code on failure
- Should ONLY be called once per process per mutex

Locking Mutexes

pthread_mutex_lock

```
int pthread_mutex_lock(pthread_mutex_t * mut);
```

- Blocks the thread's execution until the mutex has become available
- The mutex must have been initialized before use
- Returns 0 on success, error code on failure

Unlocking Mutexes

pthread_mutex_unlock

```
int pthread_mutex_unlock(pthread_mutex_t * mut);
```

- Signals that the mutex has been unlocked. Other threads may lock the mutex and continue with their execution
- Returns 0 on success, error code on failure

Destroying Mutexes

pthread_mutex_destroy

```
int pthread_mutex_destroy(pthread mutex_t * mut,
        const pthread_mutexattr_t * attr);
```

- Destroys the mutex. Should only be called once. The mutex may not be used again after destruction.
- Returns 0 on success, error code on failure

Mutex Example

- `examples/sync/mutexes.c`

A Useful End-To-End Example

http://www.thegeekstuff.com/2012/05/c-mutex-examples/