

Assignment Project Exam Help

Add WeChat powcoder

Assignment Project Exam Help  
CS:3620 Operating Systems  
<https://powcoder.com>

Add WeChat powcoder  
Condition Variables

# Another type of synchronization

- Locks allow one type of synchronization between threads – mutual exclusion

Assignment Project Exam Help

<https://powcoder.com>

- Another common requirement in multi-threaded applications – waiting and signaling
  - E.g., Thread T1 wants to continue only after T2 has finished some task
- Can accomplish this by busy-waiting on some variable, but inefficient
- Need a new synchronization primitive: condition variables

# Condition Variables

- A condition variable (CV) is a queue that a thread can put itself into when waiting on some condition

Assignment Project Exam Help

- Another thread that makes the condition true can signal the CV to wake up a waiting thread

Add WeChat powcoder

- Pthreads provide CV for user programs
  - OS has a similar functionality of wait/signal for kernel threads
- Signal wakes up one thread, signal broadcast wakes up all waiting threads

# Assignment Project Exam Help

Example: parent waits for child

```
1  int done  = 0;
2  pthread_mutex_t m = PTHREAD_MUTEX_INITIALIZER;
3  pthread_cond_t c  = PTHREAD_COND_INITIALIZER;
4
5  void thr_exit() {
6      pthread_mutex_lock(&m);
7      done = 1;
8      pthread_cond_signal(&c);
9      pthread_mutex_unlock(&m);
10 }
11
12 void *child(void *arg) {
13     printf("child\n");
14     thr_exit();
15     return NULL;
16 }
17
18 void thr_join() {
19     pthread_mutex_lock(&m);
20     while (done == 0)
21         pthread_cond_wait(&c, &m);
22     pthread_mutex_unlock(&m);
23 }
24
25 int main(int argc, char *argv[]) {
26     printf("parent: begin\n");
27     pthread_t p;
28     pthread_create(&p, NULL, child, NULL);
29     thr_join();
30     printf("parent: end\n");
31     return 0;
32 }
```

## Assignment Project Exam Help

# Why check condition in while loop?

- In the example code, why do we check condition before calling wait?
  - In case the child has already run and done is true, then no need to wait
- Why check condition with “while” loop and not “if”?
  - To avoid corner cases of thread being woken up even when condition not true (may be an issue with some implementations)

```
if(condition)
    wait(condvar)
//small chance that condition may be false when wait returns

while(condition)
    wait(condvar)
//condition guaranteed to be true since we check in while-loop
```

## Assignment Project Exam Help

# Why use lock when calling wait?

- What if no lock is held when calling wait/signal?

```
1 void thr_exit() {  
2     done = 1;  
3     Pthread_cond_signal(&c);  
4 }  
5  
6 void thr_join() {  
7     if (done == 0)  
8         Pthread_cond_wait(&c);  
9 }
```

- Race condition: missed wakeup
  - Parent checks done to be 0, decides to sleep, interrupted
  - Child runs, sets done to 1, signals, but no one sleeping yet
  - Parent now resumes and goes to sleep forever
- Lock must be held when calling wait and signal with CV
- The wait function releases the lock before putting thread to sleep, so lock is available for signaling thread

## Assignment Project Exam Help

### Example: Producer/Consumer problem

- A common pattern in multi-threaded programs
- Example: in a multi-threaded web server, one thread accepts requests from the network and puts them in a queue. Worker threads get requests from this queue and process them.
- Setup: one or more producer threads, one or more consumer threads, a shared buffer of bounded size

# Assignment Project Exam Help

## Producer/Consumer with 2 CVs

```
1  cond_t  empty, fill;
2  mutex_t mutex;
3
4  void *producer(void *arg) {
5      int i;
6      for (i = 0; i < loops; i++) {
7          Pthread_mutex_lock(&mutex);
8          while (count == 1)
9              Pthread_cond_wait(&empty, &mutex);
10         put(i);
11         Pthread_cond_signal(&fill);
12         Pthread_mutex_unlock(&mutex);
13     }
14 }
15
16 void *consumer(void *arg) {
17     int i;
18     for (i = 0; i < loops; i++) {
19         Pthread_mutex_lock(&mutex);
20         while (count == 0)
21             Pthread_cond_wait(&fill, &mutex);
22         int tmp = get();
23         Pthread_cond_signal(&empty);
24         Pthread_mutex_unlock(&mutex);
25         printf("%d\n", tmp);
26     }
27 }
```



# Assignment Project Exam Help

## Disclaimer

- *These lecture slides are based on a slide set by Youjip Won (Hanyang University) and Mythili Vutukuru (IIT Bombay)*

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder