CS 3410 Lab 10

Spring 2026

Cornell Bowers C·IS
Computer Science

Agenda

1. Spinlocks
2. Condition Variables
3. pthreads
4. A10 Tips

Spinlocks

Spinlocks

A **spinlock** is a type of lock. When a thread tries to acquire a spinlock, it "spins" until the lock becomes available.

Pseudocode

1. Check if the shared resource is available. If it isn't, repeat step 1.

1. Try to acquire the shared resource. If unsuccessful (i.e., because another thread acquired it first), repeat step 1.

LR and SC Atomic Instructions

LR (load-reserved)

- Usage: `lr.w.aqrl rd, (rs1)`
- Loads a 32-bit word from the address in register `rs1` into register `rd`
- “Reserves” the memory address at `rs1`, tracking whether any other thread writes to this address
- A “reservation” is broken if another thread writes to this address

SC (store-conditional)

- Usage: `sc.w.aqrl rd, rs2, (rs1)`
- Attempts to write the value in `rs2` to the address in `rs1`
- Only succeeds if this thread has a reservation on this address
- If successful, writes zero to `rd`; otherwise, writes a non-zero code

LR and SC are useful for implementing spinlocks in RISC-V!

Condition Variables

Condition Variables

```
1  void print_message(int* lock, int* cond, char*
                      message, int* ready) {
2    spin_lock(lock);
3    // Wait until the message is ready
4    while (!*ready) {
5      wait(lock, cond);
6    }
7    // Print the now-ready message!
8    printf("%s", message);
9    spin_unlock(lock);
10 }
```

A **condition variable** is a concurrency mechanism that allow threads to wait for some condition to become true

- A thread waits on a condition `cond` until another thread wakes it up
- A thread can broadcast to a condition `cond` to wake up all threads that are waiting on `cond`

futex

In Linux, the `futex(uint32_t* uaddr, int op, ...)` system call can be used to facilitate condition variables

- `uaddr` is the address of the condition variable
- `op` is the operation to be performed by the `futex` call (e.g., `FUTEX_WAIT`, `FUTEX_WAKE`)

pthreads

pthreads

pthreads is a Unix standard library that provides implementations for thread management, synchronization, and conditioning in C

`PTHREAD_MUTEX_INITIALIZER`	Initializes `pthread_mutex_t` lock.
`PTHREAD_COND_INITIALIZER`	Initializes `pthread_cond_t` condition variable.
`pthread_mutex_lock(pthread_mutex_t lock)`	Acquire the lock `lock`.
`pthread_mutex_unlock(pthread_mutex_t lock)`	Release the lock `lock`.
`pthread_cond_wait(pthread_cond_t cond, pthread_mutex_t lock)`	Release the lock `lock` and wait on the condition variable `cond`. On return, the calling thread is guaranteed to have reacquired `lock`.
`pthread_cond_signal(pthread_cond_t cond)`	Wakes up at least one thread waiting on the condition variable `cond`.

A11 Tips

A11 Tips

- In your `rv` container, you can call `grep "#define FUTEX_" /usr/include/linux/futex.h` to find the `futex` opcodes
- More tips can be found in the A11 instructions!