

Threads

CS 113: Introduction to C

Instructor: Saikat Guha

Cornell University

Spring 2007, Lecture 11

Processes vs. Threads

Processes ...

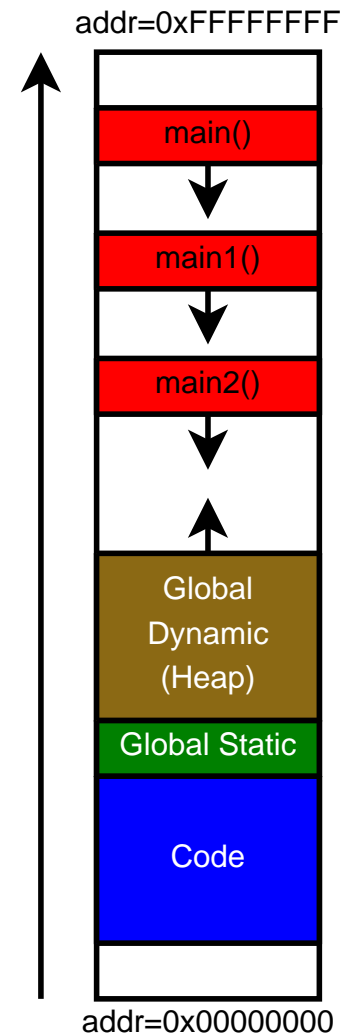
- ▶ Multiple simultaneous programs
- ▶ Independent memory space
- ▶ Independent open file-descriptors

Threads ...

- ▶ Multiple simultaneous functions
- ▶ Share the same memory
- ▶ Share the same open file-descriptors

Threads

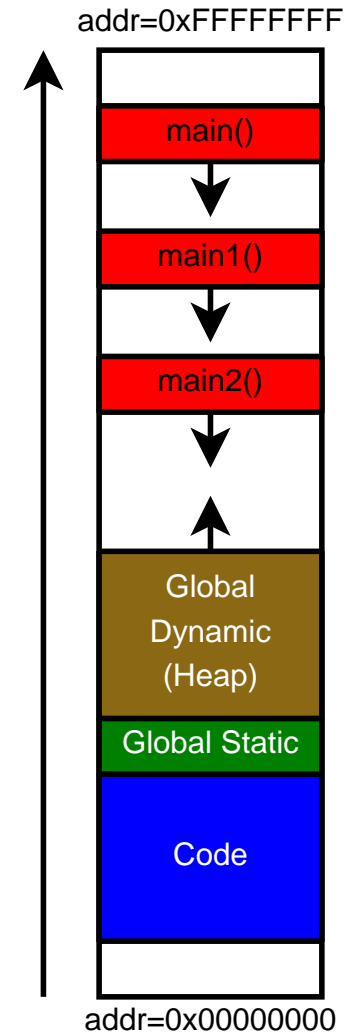
- ▶ One copy of the heap
- ▶ One copy of the code
- ▶ **Multiple** stacks



Threads

```
#include <pthread.h>
void *main2(void *arg) {
    ...
}
void *main1(void *arg) {
    ...
}
int main() {
    pthread_t id1, id2;
    pthread_create(&id1, NULL, main1, NULL);
    pthread_create(&id2, NULL, main2, NULL);
    ...
}
```

... think multiple processors (or cores)



pthread

Starting a thread

```
#include <pthread.h>
...
pthread_t id;
err = pthread_create(&id, NULL, entry_func, arg);
```

Body of a thread

```
void *entry_func(void *arg) {
    ...
```

Exiting current thread

```
    ...
    pthread_exit((void *)return_value);
}
```

pthread

Co-operative Multi-Threading on Single Processor

```
#include <sched.h>  
...  
    sched_yield()
```

- ▶ Store stack pointer, internal state etc. for current thread
- ▶ Restore stack pointer, internal state etc. for another thread
- ▶ Resume executing other thread

From the caller's perspective, `sched_yield()` blocks until the other thread calls `sched_yield()`. Allows **multiple threads to share the CPU** cooperatively.

Non-Cooperative Multi-Threading

Thread library (pthread) pre-empts thread when it invokes an OS function.

- ▶ Almost transparent when writing code
- ▶ Whole new class of bugs: Concurrency bugs
 - ▶ Multiple threads accessing same object concurrently
 - ▶ Solution: Locks – only one thread can grab lock
 - ▶ More of this is CS414/415