Faculty of Engineering and Applied Science

SOFE - 3950U Operating Systems

Winter 2023

Tutorial 5 - POSIX Thread

Vishan Patel - 100784201
Akshat Kapoor - 100781511
Steven Mai - 100781485
Evidence Okeke - 100755328

Wednesday, March 8th, 2023

CRN 74171, Group 2

Github Link: https://github.com/23Vishan/OS-Tutorial-5/tree/main

Conceptual Questions

1. Read the pthread documentation and explain the following three functions

pthread_create(thread, attr, start_routing, args)
Starts a new thread in the calling process. The new thread begins by invoking the *start_routine()* and passes *arg* as the argument. Attributes that are stated by *attr* are used to specify behaviour different from default.

pthread_join(thread, retval)
Waits for a thread to terminate. Once the thread terminates, it then detaches it. If *retval* was not null, then the exit status of the thread is returned.

pthread_exit(retval)
Terminates the thread. Returns the exit status through *retval* that is available to other threads in the same process.

2. Explain how the memory of threads work in comparison to processes, do threads share the same memory, can threads access the memory of other threads?

Threads execute their code in the same memory space while processes do not. Usually there are multiple threads in a process. As long as the threads are in the same process, they should be able to access one another's memory.

3. Name the differences between multithreading and multiprocessing (multiple processes). What are the advantages and disadvantages of each?

	Multithreading	**Multiprocessing**
Differences	- Allows individual process to have many code segments - Threads run parallel - Works with individual process and computes threads - Shares code, data, and files but not register or stack	- Has more than two processors - Improves a system's reliability - Helps increase computing power - Does not share code, data, files, register, or stack
Advantages	- Lightweight - Faster than processes when starting and switching tasks	- Gets more work done in shorter period - Straightforward

	- Share address space and process memory pool	- Can interrupt and kill child processes - Saves money compared to single processors
Disadvantages	- Can not interrupt - More difficult to understand	- Has large memory footprint

4. Provide an explanation of mutual exclusion, what is a critical section?

Mutual exclusion is when simultaneous access of a shared resource is prevented by a program object. Critical section is the part in a code that gets accessed frequently, simultaneously, and changes values of global variables. Threads that in the critical section may produce different outputs if mutual exclusion is not implemented.

5. Research the functions used to perform mutual exclusion with pthreads and explain the purpose of each function.

pthread_mutex_lock()
The first thread to reach this code section locks it down. If another thread reaches this part and it is locked, the thread must wait until the section becomes unlocked. Only lets one thread access the critical section at a time.

pthread_mutex_unlock()
Unlocks the code section. Used when a thread is done with a critical section.

pthread_barrier_init()
Is a synchronisation object. Blocks threads from preceding any further until the specified amount of threads have called *pthread_barrier_wait()*.

pthread_barrier_wait()
Threads stop at this function and may only proceed when the number of threads specified in *pthread_barrier_init()* have been met.