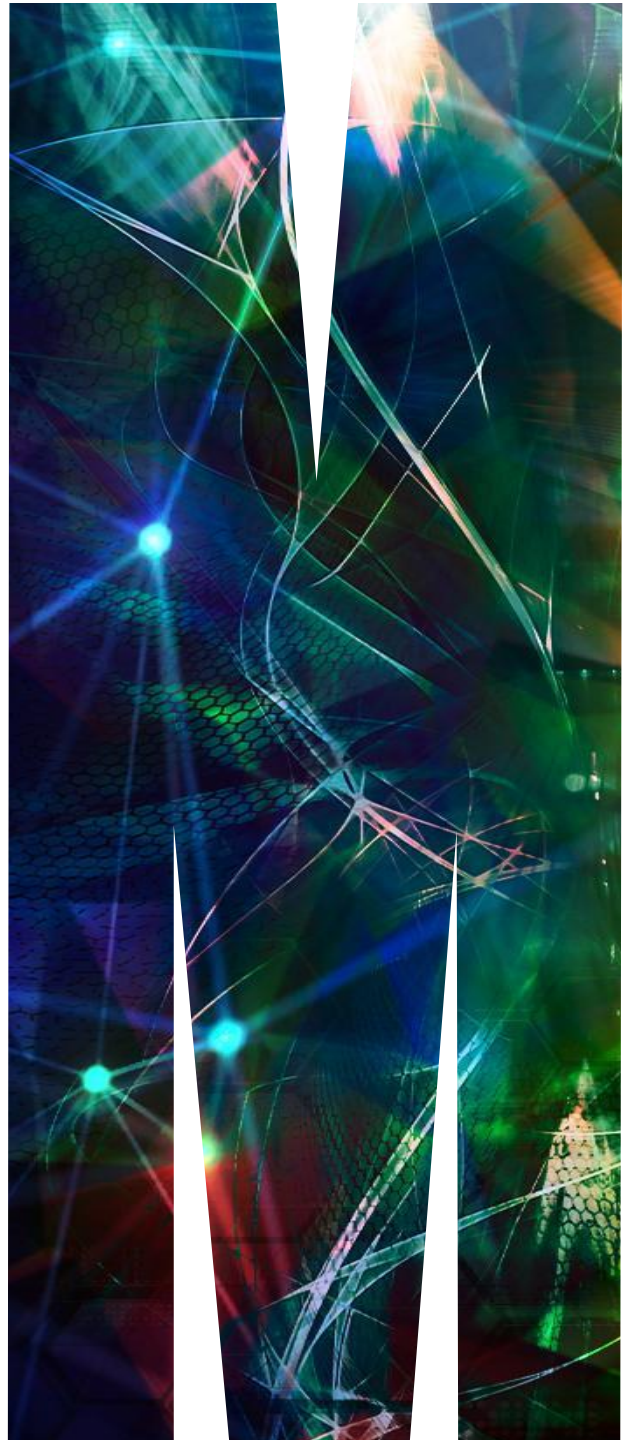
MONASH University

FIT2100 - OPERATING SYSTEMS

WEEK 7 - WORKSHOP 06

A/Prof. Campbell Wilson
Dr. Adamu Muhammad Buhari
Dr. Charith Jayasekara

Faculty of Information Technology



CONTENT

WORKSHOP TOPICS

- The Thread Model, Usage and Implementation
- POSIX Threads
- Concurrency Problems
- Race Condition
- Mutual Exclusion

Week 7 Learning Objectives

Threads and Concurrency

- Understand the distinction between process and thread
- Describe the basic design issues for threads
- Explain the difference between user-level threads and kernel-level threads
- Discuss the basic concepts related to concurrency
- Understand the concept of race condition
- Describe the mutual exclusion requirements

RECAP TIME - THREADS

MULTIPLE CHOICE / DISCUSSION QUESTIONS (10 mins)

PollEV Time - Engage and Interact

THREADS

THREAD EXECUTION TIME (3 MINS)

Given a single-threaded application that takes 10 seconds to execute, if the same application is re-implemented with 5 threads and each thread executes 20% faster due to parallelism, how long will it take for the multithreaded application to execute in **a single threaded uniprocessor system**?

THREADS

THREAD CREATION OVERHEAD (3 MINS)

Suppose creating a user-level thread takes 10 microseconds, while creating a kernel-level thread takes 100 microseconds. If a program creates 100 threads of each type, calculate the total time spent just on thread creation.

THREADS

THREAD SWITCHING (5 MINS)

Given that a system takes 20 microseconds to switch between user-level threads and 200 microseconds to switch between kernel-level threads, how long will it take to context switch between threads 1000 times in each scenario?

You can assume the system switches between same level threads in this scenario

THREADS

THREAD EFFICIENCY (10 MINS)

Imagine you're examining the performance of a quad-core CPU, such as the older versions of Intel Core i7, when confronted with a CPU-bound task. This task is divided into four threads, intended to execute concurrently on each of the available cores. However, you've discovered an additional 10% overhead introduced due to the multithreading process itself, which affects the overall execution time of each thread. Each thread is assigned an equal portion of the task.

What is the speedup achieved from multithreading when compared to a single-threaded system? Assume that each thread processes the same amount of the task.

The speedup, S, is calculated as:

$$S = \frac{\text{Time taken without multithreading}}{\text{Time taken with multithreading}}$$

RECAP TIME - CONCURRENCY

MULTIPLE CHOICE / DISCUSSION QUESTIONS (20 mins)

PollEV Time - Engage and Interact

RACE CONDITIONS

RACE CONDITION CALCULATION (15 MINS)

A program has two threads, Thread A and Thread B. The program uses a shared counter variable, initially set to 0.

Thread A's operation:

- Read the counter.
- Increment the read value by 2.
- Store the incremented value back to the counter.

Thread B's operation:

- Read the counter.
- Double the read value.
- Store the doubled value back to the counter.

Both threads perform their respective operations 2 times without any synchronization mechanisms. What could be some possible final values of the counter?

CONCURRENCY

CONCURRENCY CONTROL (10 MINS)

A web hosting server is configured with 20 units of memory. It can handle 100 simultaneous users when running a lightweight website, consuming 10 units of memory without any contention issues. However, when a complex web application with database interactions is hosted, the server experiences memory contention. For every user above the initial 100 users, an additional two units of memory (in addition to the 0.1 Units/user) are required due to the contention caused by increased database interactions. Given the remaining 10 units of memory available after serving the initial 100 users, calculate the maximum number of users the server can handle when running the complex web application without running out of memory.

MONASH University

Summary

- Topics covered
  - The Thread Model, Usage and Implementation
  - POSIX Threads
  - Concurrency Problems
  - Race Condition
  - Mutual Exclusion

- Next week
  - Advance concurrency