CS140 Midterm Review

Administrivia

Midterm is **in class** on Wednesday

- Open notes
- No textbook!
- No electronic devices!
- Can **print** any material you want (bring lecture notes!)

50% of class grade from exams:

max(midterm > 0 ? final : 0, (midterm + final)/2)

Material Covered

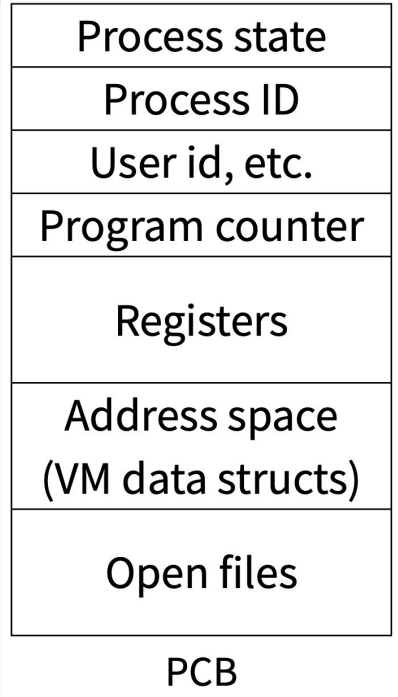
- Threads and Processes
- Concurrency
- Scheduling
- Virtual Memory Hardware
- Virtual Memory OS Techniques
- Synchronization
- Linking
- Memory allocation (Dawson’s Monday lecture)

Threads and Processes

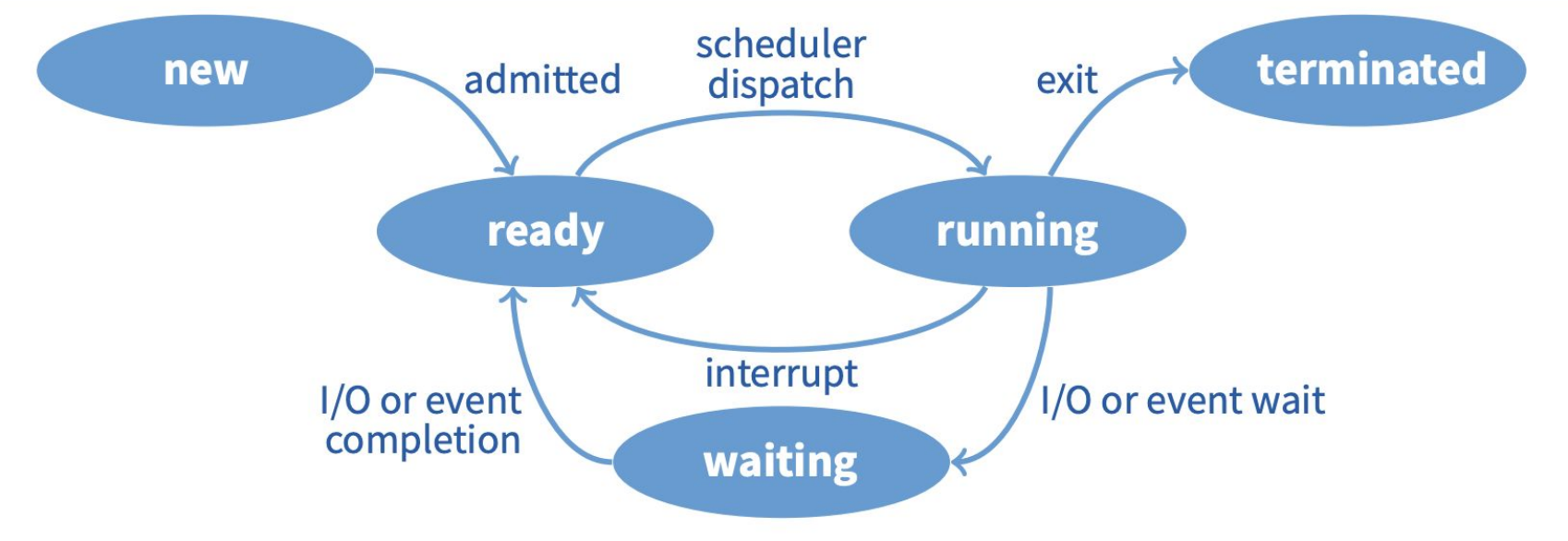
A **process** is an instance of a program running.

The **Process Control Block (PCB)** stores information about each process.

During a **context switch**, save state into PCB (registers, address spaces) and reload state from the next PCB.



Threads and Processes



Threads and Processes

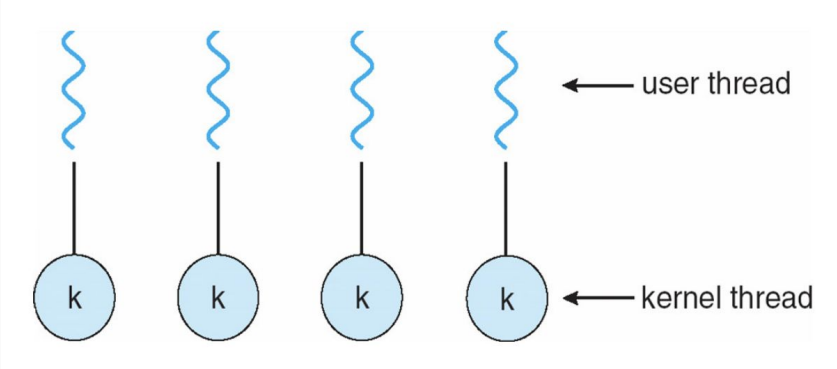
A **thread** is a schedulable execution context.

- Most popular abstraction for concurrency
- Lighter weight abstraction than processes
- Allows one process to use multiple CPUs/cores
- Allows programs to overlap I/O and computation

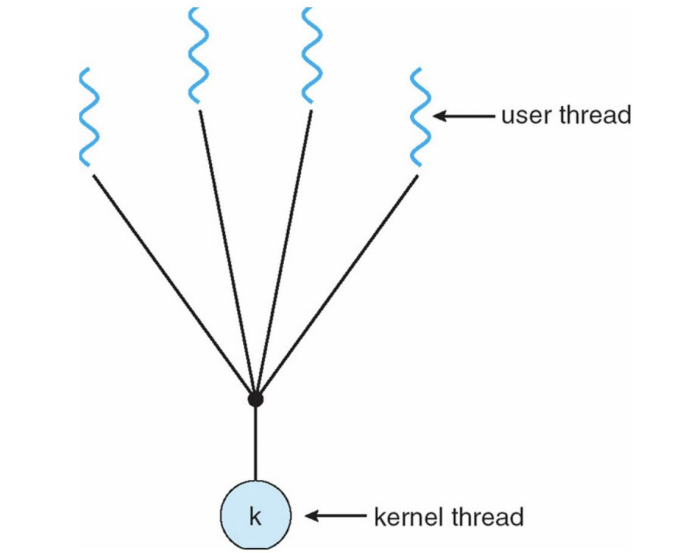
Kernel threads can take advantage of multiprocessor, but are heavy-weight

User threads are more light-weight and flexible, but can't take advantage of multiple CPUs

Threads and Processes



Kernel threads



User threads

Concurrency

Prevent **data races** by defining **critical sections** (require mutual exclusion, progress, and bounded waiting).

Synchronization primitives:

1. **Locks:** useful for mutual exclusion, put lock around code if you want to make it atomic
2. **Condition variables:** generally used to avoid busy waiting, use wait and signal
3. **Semaphores:** typically used with a shared resource that changes availability based on an integer number

Question #1

A and B are initialized to 0. What are the final values of A and B?

Thread #1:

```
while (A == 0);
B = 1;
```

Thread #2:

```
while (B == 0);
A = 1;
```

Question #1

A and B are initialized to 0. What are the final values of A and B?

Thread #1:

```
while (A == 0);
B = 1;
```

Thread #2:

```
while (B == 0);
A = 1;
```

*Depends on your **memory model**!*
If sequential consistency: deadlocks (A = 0, B = 0)
Under a different memory model: possible to complete (A = 1, B = 1) or deadlock

Concurrency

Sequential consistency: The result of execution is as if all operations were executed in some sequential order, and the operations of each processor occurred in the order specified by the program.

If you use synchronization primitives properly, behavior should be indistinguishable from sequential consistency.

Scheduling

Scheduling algorithms try to optimize for throughput, turnaround time, response time, CPU utilization, and waiting time:

- FIFO
- Shortest Job First
- Round-robin
- Priority scheduling
- Multilevel feedback queue

Question #2

Describe an advantage and disadvantage for both SJF and round-robin scheduling.

Question #2

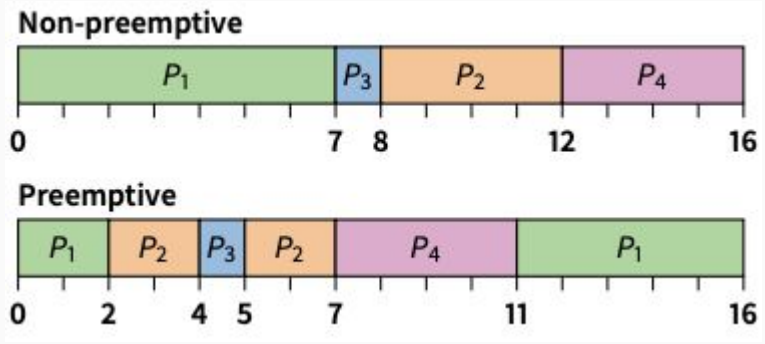
Describe an advantage and disadvantage for both SJF and round-robin scheduling.

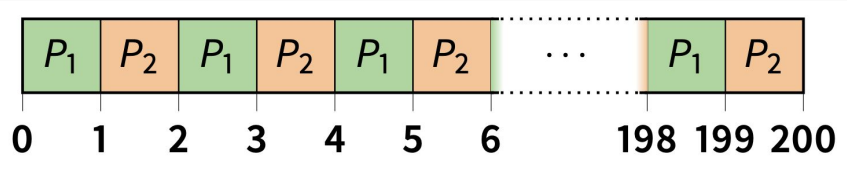
SJF

+ minimize average waiting time
- can lead to unfairness or starvation

Round-robin

+ fairness and no starvation
- high average turnaround time





Virtual Memory HW

OS gives each process its own virtual address space.

Segmentation: divide memory into segments; keep track of base and bound registers

- Causes **external fragmentation:** enough total memory, but not contiguous so cannot be used

Paging: map virtual pages to physical pages

- Causes **internal fragmentation:** memory block assigned to process is larger than needed

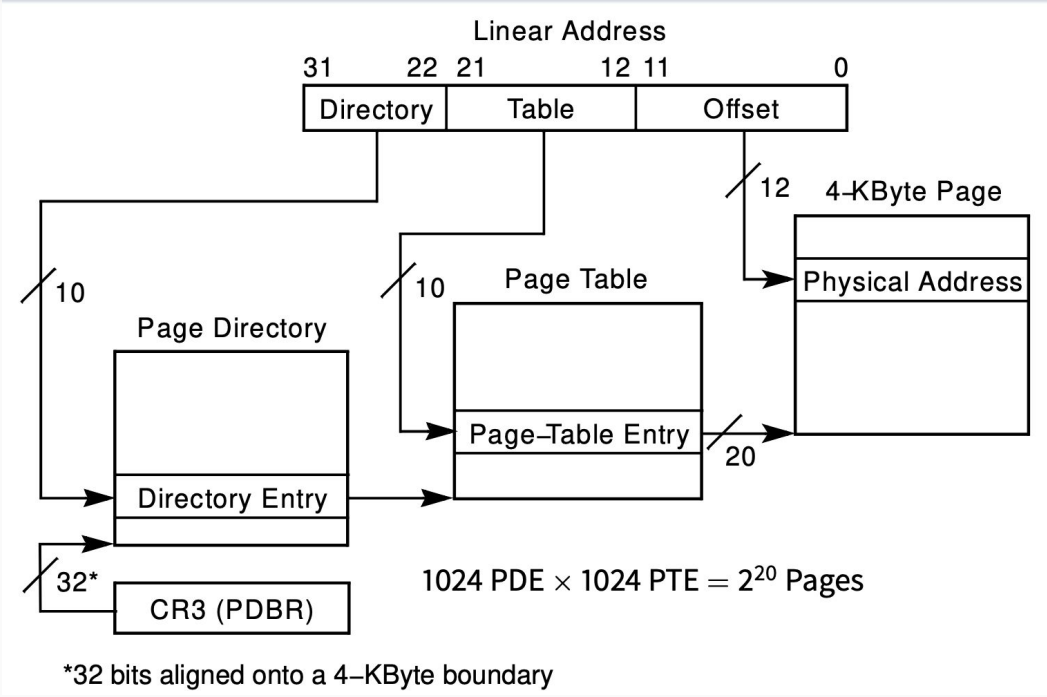
Virtual Memory

Have more virtual memory than physical memory: save unused virtual pages to disk.

Thrashing: constant paging because **working set** cannot be in memory at once

- Working set model: process can be in memory iff all pages used by process can be in memory

Virtual Memory

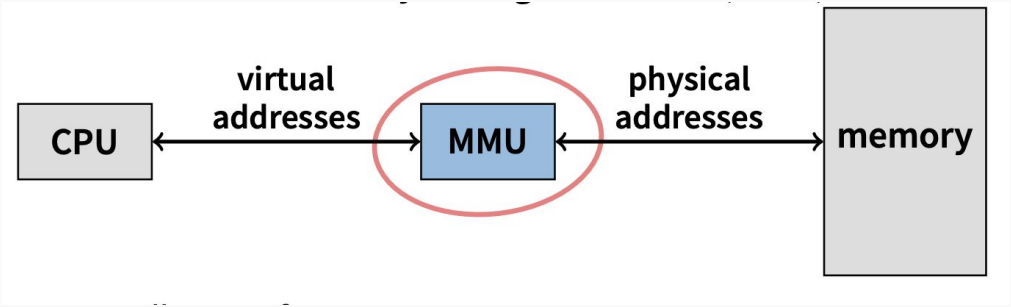


Virtual Memory

Memory Management Unit **(MMU)**: hardware to translate from virtual to physical addresses

Translation Lookaside Buffer **(TLB)**: cache of recently used translations

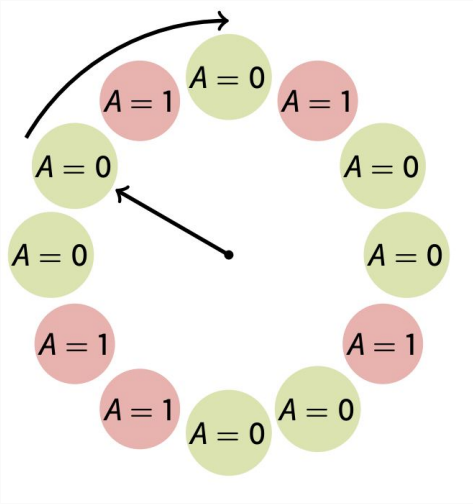
- What to do on a context switch flush TLB or tag each entry with associated process



Virtual Memory

Page replacement algorithms: decide which page to write to disk

- Clock algorithm: second-chance replacement
- Random eviction: avoids pathological cases
- LFU (least frequently used)
- MFU (most frequently used)



Question #3

True or False: Increasing the page size will likely decrease the working set size (measured in bytes).

Question #3

True or False: Increasing the page size will likely decrease the working set size (measured in bytes).

- *False*

Question #3

True or False: Increasing the page size will likely decrease the working set size (measured in bytes).

- *False*

True or False: Increasing the page size will likely decrease the chance that page revocation requires a disk write.

Question #3

True or False: Increasing the page size will likely decrease the working set size (measured in bytes).

- *False*

True or False: Increasing the page size will likely decrease the chance that page revocation requires a disk write.

- *False: As page size increases, the likelihood that a page is dirty (and so needs to be written to disk on eviction) increases.*

Synchronization

Coherence: concerns access to a single memory location (must obey program order if access from only one CPU)

- MESI protocol

Consistency: concerns ordering across memory locations (different CPUs can see the same write happen at different times)

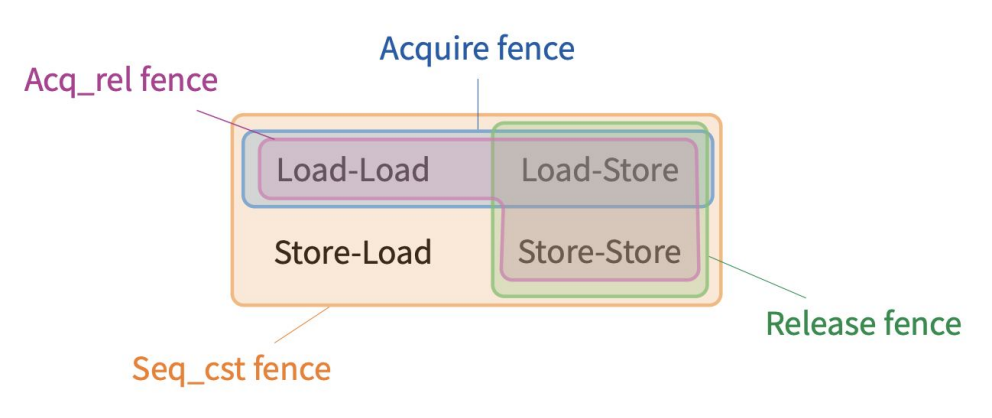
- Sequential consistency matches our intuition (as if operations from all CPUs interleaved on one CPU)

Synchronization

A happens before B (A → B) when:

- A is sequenced before B (in the same thread)
- A synchronizes with B
- A is dependency-ordered before B (instruction B depends on A if B uses results of A → memory_order_consume)
- There is another operation X such that A → X → B

Synchronization



X-Y fence = operations of type X sequence before the fence happen before operations of type Y sequenced after the fence. (Still confused? Read this)

Question #4

If you care more about speed than accuracy when incrementing some counter (e.g. number of hits to a website), which of the below memory_order values should be used when incrementing?

A. memory_order_seq_cst (full sequential consistency)
B. memory_order_relaxed (no memory ordering)
C. Not needed

Question #4

If you care more about speed than accuracy when incrementing some counter (e.g. number of hits to a website), which of the below memory_order values should be used when incrementing?

A. memory_order_seq_cst (full sequential consistency)
B. memory_order_relaxed (no memory ordering)
C. Not needed

No such thing as a benign race!

Synchronization

Read-Copy-Update (RCU): some data is read much more often than written, so optimize for the common case of reading

- Update by making copy and swapping pointer
- Need dependency ordering (memory_order_consume, no op except on alpha)

MCS lock: reduces bus traffic on cache-coherent machines, improves fairness (NUMA)

Synchronization

All of the following conditions necessary for deadlock:

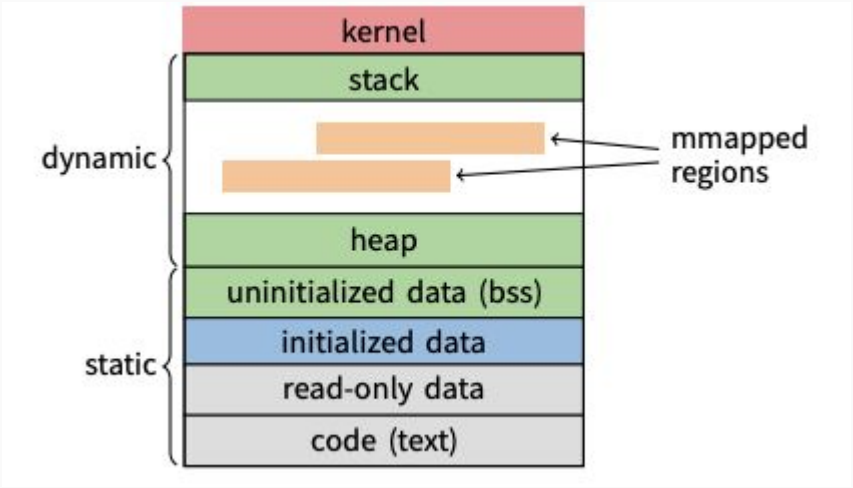
1. Limited access (mutual exclusion)
2. No preemption
3. Multiple independent requests (hold and wait)
4. Circularity in graph of requests

Prevent deadlock with partial order on resources

Linking

Who builds what?

- **Heap:** allocated and laid out at runtime by malloc
- **Stack:** allocated at runtime, layout by compiler
- **Global data/code:** allocated by compiler, layout by linker
- **Mmapped regions:** managed by programmer or linker



Linking

What does the linker do?

1. Collect together all pieces of a program
2. Coalesce like segments
3. Fix addresses of code and data so the program can run

2 passes:

1. Coalesce like segments, construct global symbol table, compute virtual address of each segment
2. Patch references using file and global symbol table

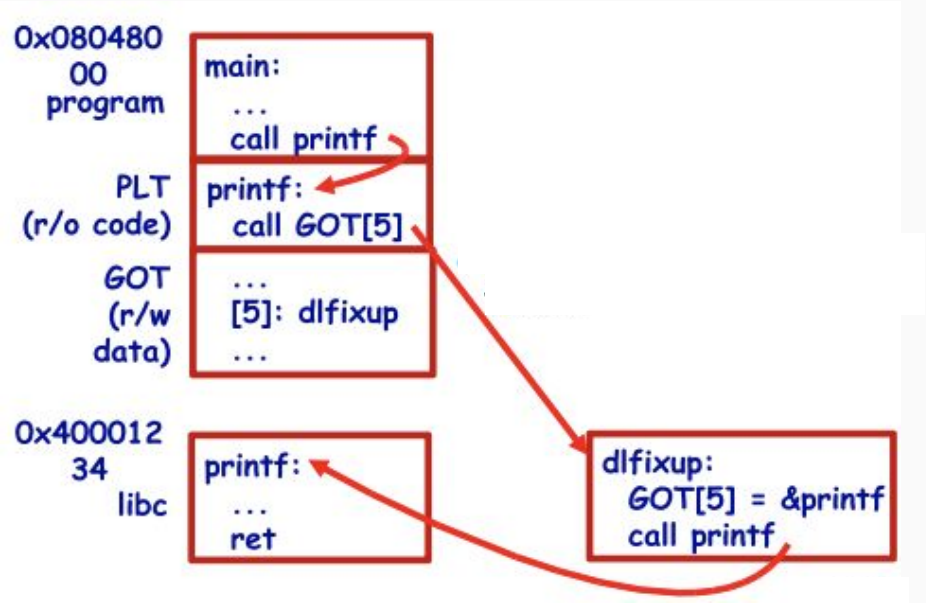
Linking

Lazy dynamic linking

PLT = Procedure Lookup Table (r/o)

GOT = Global Offset Table (r/w)

(Still confused? Read this)



Memory Allocation

Covered in Monday’s lecture!

General Tips

Old exams won’t necessarily cover the same material or have the same format.

Print and bring lecture slides

… but know the lecture material (you won’t have the time to read through the slides to understand concepts)

David’s office hours today at 4PM

Good luck!