Lecture 7.1: OS Abstractions Processes

EN 600.320/420

Instructor: Randal Burns

19 February 2018

Department of Computer Science, *Johns Hopkins University*

Terminology

- Concepts
  - Task: sequence of instructions that operate as a group
  - Unit of execution (UE): process or a thread, the execution context for a task
  - Processing Element (PE): hardware element that runs the UE
- For Java these are going to be
  - Runnable object
  - Thread
  - Processor core
- *What is the relationship between the number of UEs and number of PEs (threads and cores)?*

What's a process?

- "A process is the operating system's abstraction for a running program" (Bryant and O'Halloran, Computer Systems)
  - Processes "provide the illusion that the program is the only one running on the system"
- Each process appears:
  - To have exclusive use of the hardware
  - To execute instructions one after another without interruption
- By definition, processes do not share memory
- The process abstraction allows for multiple serial programs to run concurrently and in parallel

Context

- **Context switch:** operating systems transparently switch among the running processes
- **Context** is the per process state maintained by the OS
  - Needed to suspend and resume processes
  - Includes: program counter, register file, contents of memory

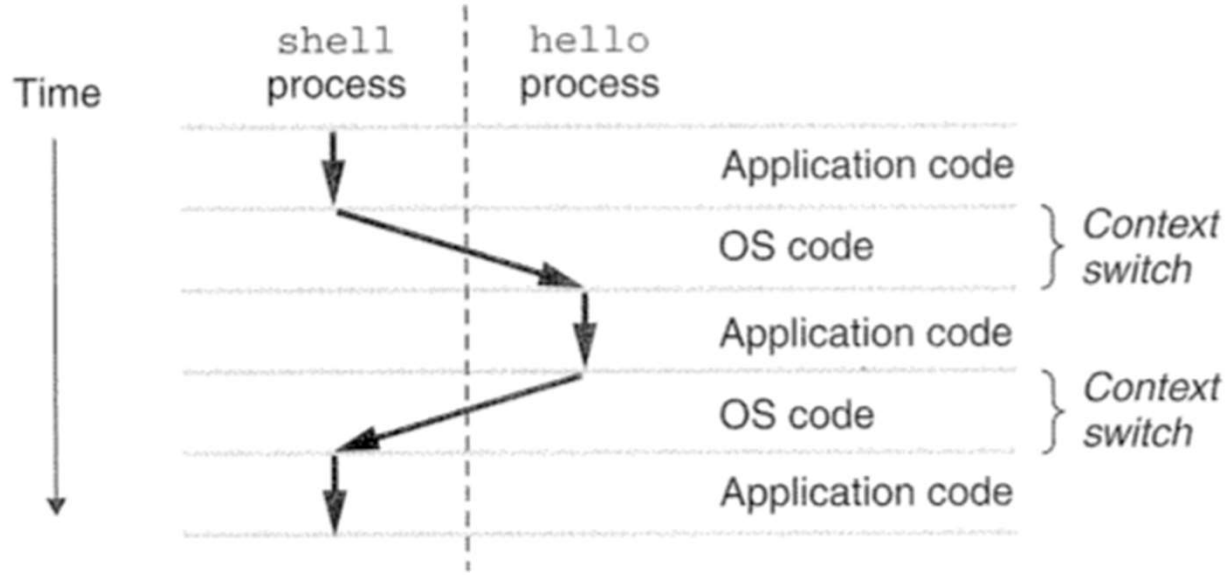
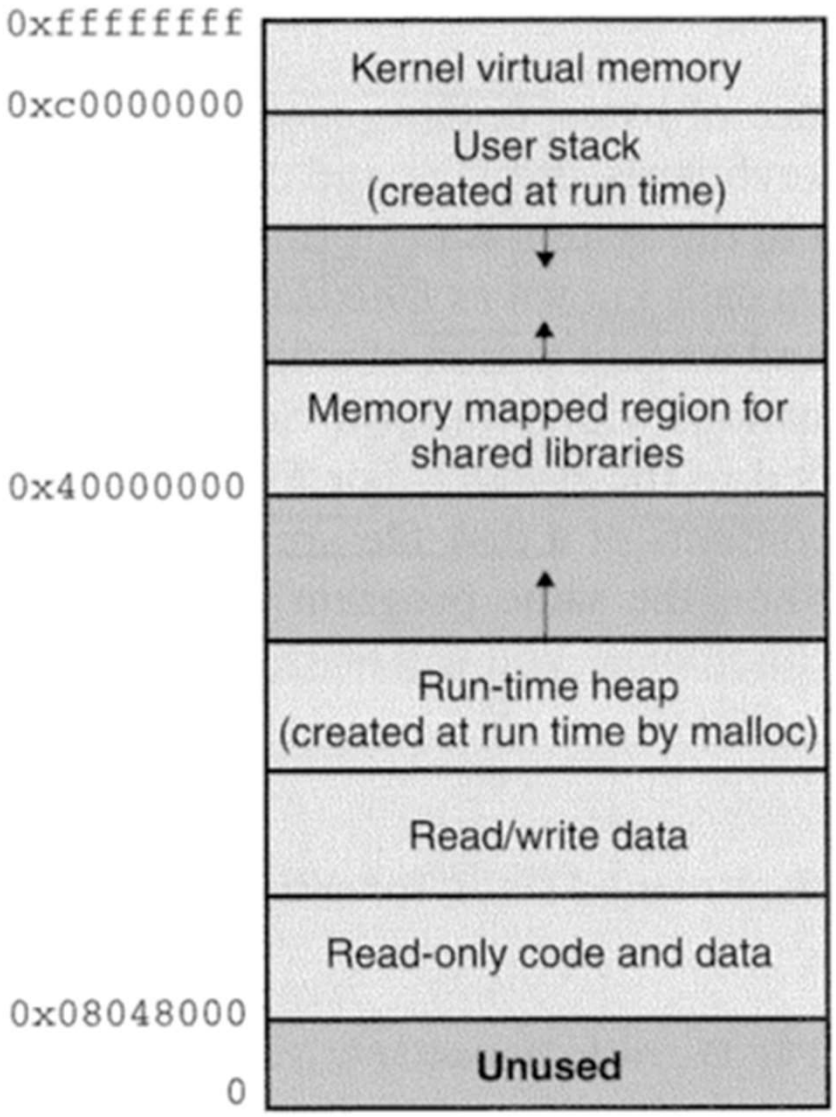


Figure 1.12 **Process context switching.**

Virtual Memory

- Processes appear to have exclusive use of memory
  - It's actually a virtual address space, because the addresses don't correspond to HW addresses
- Memory contains the state of a single program
  - Code, data, heap, stack
  - Memory mapped (shared) libraries



Parallel Programming w/ Processes

- Flipping coins with fork() and exec()
  - fork() creates a copy of the running processs
  - Exec() loads a new program into that process

```
…
for ( $i=0; $i < num_cores; $i++ )
{
  if (fork()==0) { exec PPCoinFlip ($num_flips/$num_cores); }
}
…
```

- A parallel implementation calls multiple instances
  - Task: flip coin N times
  - UE: process
  - PE: core
- OS assigns processes to cores
  - That's its job, operate the system, manage H/W resources

Advantages/Disadvantages of Processes

- No shared state!
- Applications that need to share state, must do so explicitly using inter-process communication (IPC)
  - E.g. RPC, sockets, shmem (shared memory area).
  - All interfaces are clunky.
- No shared state, means no dependencies
  - Easy to parallelize on message passing architectures
- Simplest parallel program
  - Run the same program on lots of nodes
  - SPMD (embarassibly parallel) programs

```
> mpirun PPCoinFlip (10000000);
```