CSE 410: Systems Programming
Midterm Review

Ethan Blanton

Department of Computer Science and Engineering
University at Buffalo
POSIX and C Summary

- C is statically typed.
- C exposes many architecture details.
- C has no garbage collector, constructors, or destructors.
- The POSIX API is based on UNIX.
- POSIX provides an interface to the OS kernel.
- A C string is an array of characters.
- C and POSIX provide a rich text-based I/O API.
- Pointers allow direct access to memory by address.
- The “C compiler” is actually a chain of operations.
Memory Representation Summary

- Machines use **words** for memory and register access
- **Hexadecimal** is convenient for representing words on modern systems
- **C structures are** **C datatypes laid out adjacent in memory**
- Word sizes have **alignment implications** on memory layout
- Integer representation has complications!
- Floating point representations have different complications!
A program is code that can be executed, a process is that code running on a system.

The linker joins multiple objects into an executable.

A loader prepares a program that has been copied into memory for execution.

Program code (text), initialized data (data), and uninitialized data (bss) are present in both a program and a process.

The heap and stack can both grow, the former “upward” toward higher addresses and the latter “downward” toward lower addresses.
The kernel manages shared resources.

Userspace and the kernel are in different protection domains.

Processes request services from the kernel using system calls.

UNIX processes are created with `fork()`.

The `exec()` system call loads a new program.

The kernel manages other state for processes, such as:
- The current directory
- Environment variables
- Open files
Input and Output Summary

- UNIX I/O is defined by the POSIX Standard
- Standard I/O is defined by the C Standard
- The kernel tracks open files with file descriptors
- All file I/O goes through the kernel
- The standard I/O library is buffered
Pipes and Redirection Summary

- Pipes form a UNIX IPC mechanism.
- They are a kernel communication channel that provides file semantics.
- Pipes have finite buffer space.
- File descriptors are indirect pointers to open files.
- Fork copies file descriptors and thus open file state.
- File descriptors can be explicitly copied with dup() and dup2().
Virtual Memory Summary

- **Virtual memory:**
  - uses a *memory management unit*
  - allows the CPU to operate in a *virtual address space* that may be different from the *physical address space*
  - the MMU translates virtual addresses to physical addresses

- **Paging** is a common model for virtual memory.
- Paged systems break *both address spaces* into *pages*.
- Pages can be *mapped individually* between virtual and physical addresses.
- **Page tables** allow the MMU to translate addresses.
- **Page faults** bring mapped but unallocated pages into memory.
License

Copyright 2018 Ethan Blanton, All Rights Reserved.

Reproduction of this material without written consent of the author is prohibited.

To retrieve a copy of this material, or related materials, see https://www.cse.buffalo.edu/~eblanton/.