RECITATION 7: SIGNALS AND I/O

15-213-M12

Rick Benua

Today:

- Shell Lab
- Midterm Exam
- Signals
- Process Control
- Unix I/O

Shell Lab

- Due Tuesday
- Testing
  - Test Suite
  - Race conditions

Midterm Exam

- Local students: In class next Thursday (7/12)
- Distance students: At a test center (or self-proctored) between 7/12 and 7/14
- Coverage:
  - Integers / Bitwise operations
  - Floating Point
  - Structures / Alignment
  - Assembly / Stack Discipline
  - Caching
  - Signals
  - Unix I/O

Signals

- Communication from the OS to the program
- Received on context switch
- Responses – program dependent, changed with signal() or sigaction()
  - Ignore
  - Terminate
  - Suspend (or resume on SIGCONT)
  - Execute signal handler
- Default action varies by signal
- SIGSTOP, SIGKILL cannot be changed
  - Reliable way to get rid of an unresponsive process

Signals – Blocking

- Signals can be “blocked” by a process
- Blocked signals are not ignored, but are not received until the signal is unblocked
- Blocked (or not-yet-received) signals are not queued – handler executes once no matter how many times the signal was sent

Important Signals

- SIGINT
  - By default, terminates the process
  - Can be handled to clean up resources
  - Ctrl+C sends SIGINT to terminal's active process group
- SIGTSTP
  - By default, suspends process
  - Ctrl+Z sends SIGTSTP to terminal's active process group
- SIGKILL
  - Always terminates process
  - OS sends SIGTERM, then later SIGKILL to all processes on shutdown
  - "kill -9"
- SIGCHLD
  - Received by parent process when child terminates or suspends
  - Ignored by default

Process Control

- Process IDs
- fork()
  - Creates a new process
  - Called once, returns twice
  - Returns 0 to child, child's PID to parent
- Process Groups
  - Inherited on fork()
  - Can be changed with setpgid()
- execve()
  - Current process starts executing a different executable file
  - Uses current address space, system resources
  - Called once, never returns

Process Control

- kill()
  - Sends a signal to a process
  - Must be owned by the same user
- wait(), waitpid()
  - Cleans up terminated process resources
  - By default, waits for process to terminate
  - Can set WNOHANG option
  - status argument to waitpid() gives calling process information about the terminated process
  - See “man waitpid” for details
- Terminated processes must be “reaped” by wait
  - init (PID 1) reaps processes if they are reparented to it (if their parent dies)

Unix I/O

- File Descriptors
  - Small integer identifying a file in use by the process
  - Each process has a table mapping file descriptors to OS-level file structures (stores mode, offset, etc)
  - File structures shared per process, descriptors not
  - Returned by open(), dup(), etc
  - Duplication (dup, dup2) only duplicates file descriptor, not file structure
  - Pass file descriptors to read(), write(), etc
  - Close files when done – leaking file descriptors over the course of a long running program (like a shell!) is bad
  - Automatically cleaned up on exit from program