



Operating System

Inter-Process Communication

IPC

- ◆ How does one process communicate with another process?
 - semaphores -- signal notifies waiting process
 - software interrupts -- process notified asynchronously
 - pipes -- unidirectional stream communication
 - message passing -- processes send and receive messages.



Software Interrupts

- ◆ Similar to hardware interrupt.
- ◆ Processes interrupt each other (often for system call)
- ◆ Asynchronous! Stops execution then restarts
 - ctrl-C
 - child process completes
 - alarm scheduled by the process expires
 - ◆ Unix: SIGALRM from alarm() or setitimer()
 - resource limit exceeded (disk quota, CPU time...)
 - programming errors: invalid data, divide by zero



Software Interrupts

- ◆ SendInterrupt (pid, num)
 - type num to process pid,
 - kill() in Unix
- ◆ HandleInterrupt (num, handler)
 - type num, use function handler
 - signal() in Unix
- ◆ Typical handlers:
 - ignore
 - terminate (maybe w/core dump)
 - user-defined
- ◆ (Hey, show demos!)



Unreliable Signals

```

◆ Before POSIX.1 standard:
signal(SIGINT, sig_int);
...
sig_int() {
    /* re-establish handler */
    signal(SIGINT, sig_int);
}

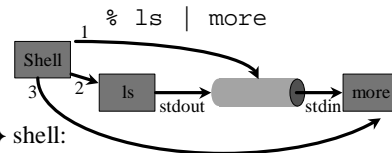
```

- ◆ Another signal could come before handler re-established!



Pipes

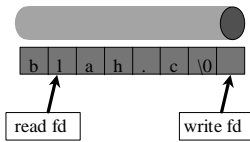
- ◆ One process writes, 2nd process reads



- ◆ shell:
 - 1 create a pipe
 - 2 create a process for ls command, setting stdout to write side of pipe
 - 3 create a process for more command, setting stdin to read side of pipe



The Pipe



◆ Bounded Buffer

- shared buffer (Unix 4096K)
- block writes to full pipe
- block reads to empty pipe



The Pipe

- ◆ Process inherits file descriptors from parent
 - file descriptor 0 stdin, 1 stdout, 2 stderr
- ◆ Process doesn't know (or care!) when reading from keyboard, file, or process or writing to terminal, file, or process
- ◆ System calls:
 - read(fd, buffer, nbytes) (`scanf ()` built on top)
 - write(fd, buffer, nbytes) (`printf ()` built on top)
 - pipe(rfd) creates a pipe
 - ◆ rfd array of 2 fd. Read from rfd[0], write to rfd[1]
- ◆ (Hey, show sample code!)

