I/O Multiplexing, Signals

CS 241

April 10, 2014

University of Illinois

epoll() example

see epoll.c

Signals

Signals

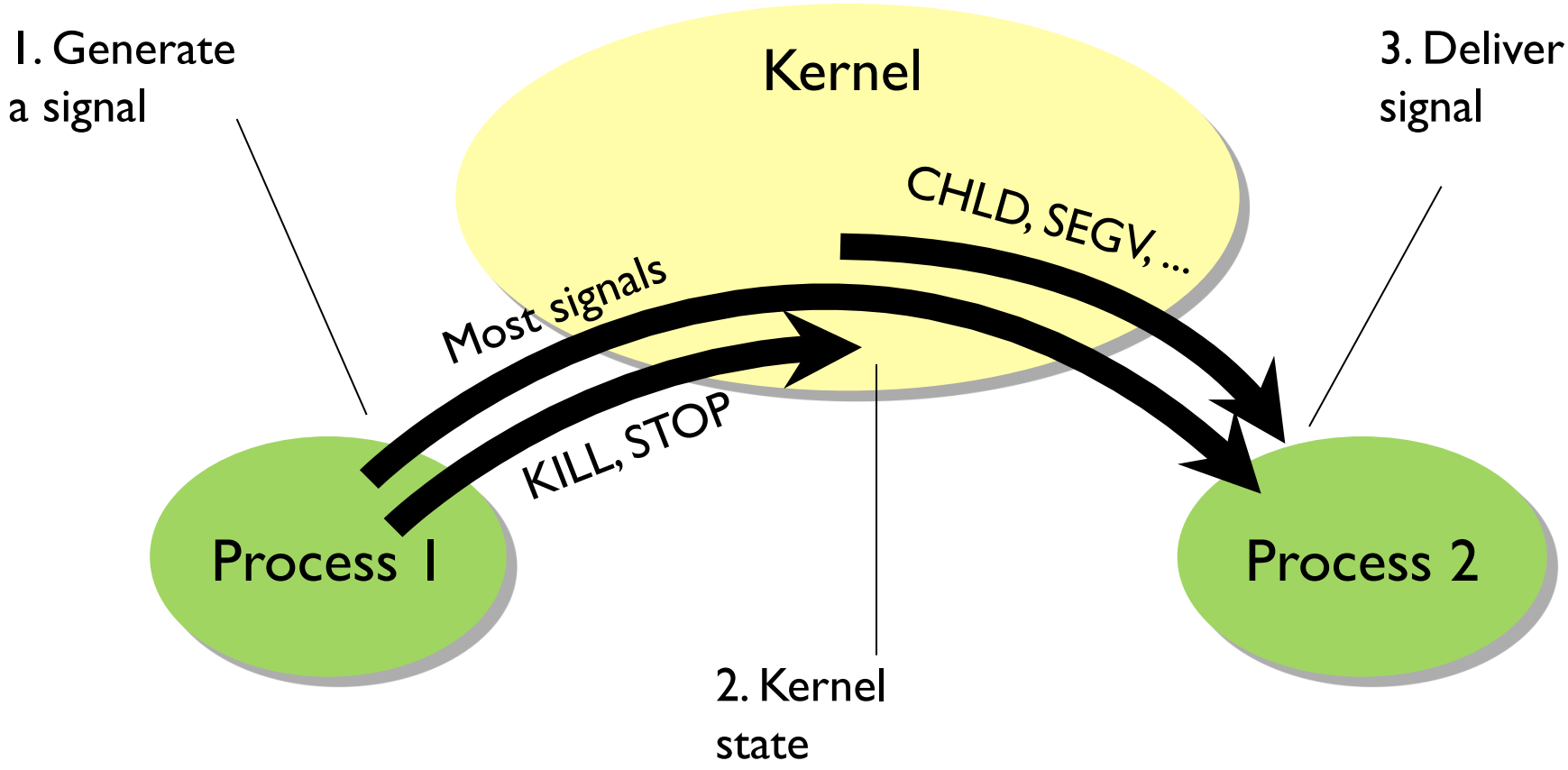
A signal is an asynchronous notification of an event

- Asynchronous: could occur at any time
- Interrupts receiving process; jumps to signal handler in that process
- A (limited) menu of event types to pick from

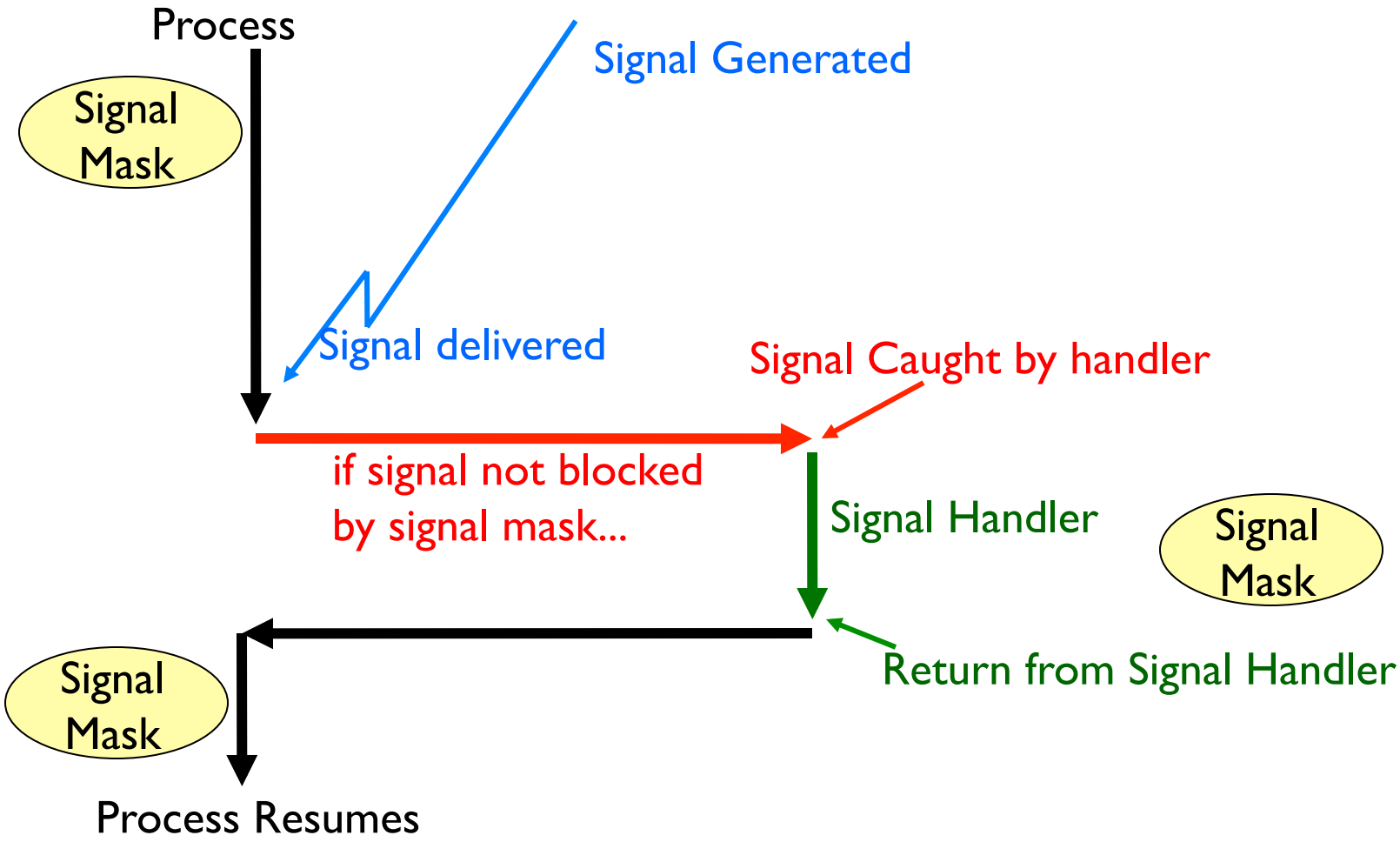
What events could be asynchronous?

- Email message arrives on my machine
  - Mailing agent (user) process should retrieve it
- Invalid memory access
  - OS should inform scheduler to remove process from the processor
- Alarm clock goes off
  - Process which sets the alarm should catch it
- Program's configuration file is modified
  - Program should reload its files

Signaling overview



Signaling: Inside Process 2



Example: Catch SIGINT

see signal_handler.c

Some POSIX signals (see signal.h)

NAME	Default Action	Description
SIGHUP	terminate process	terminal line hangup
SIGINT	terminate process	interrupt program
SIGQUIT	create core image	quit program
SIGILL	create core image	illegal instruction
SIGTRAP	create core image	trace trap
SIGABRT	create core image	abort(3) call (formerly SIGIOT)
SIGEMT	create core image	emulate instruction executed
SIGFPE	create core image	floating-point exception
SIGKILL	terminate process	kill program
SIGBUS	create core image	bus error
SIGSEGV	create core image	segmentation violation
SIGSYS	create core image	non-existent system call invoked
SIGPIPE	terminate process	write on a pipe with no reader
SIGALRM	terminate process	real-time timer expired
SIGTERM	terminate process	software termination signal
SIGURG	discard signal	urgent condition present on socket
SIGSTOP	stop process	stop (cannot be caught or ignored)
SIGTSTP	stop process	stop signal generated from keyboard
SIGCONT	discard signal	continue after stop

Some POSIX signals (see signal.h)

NAME	Default Action	Description
SIGCHLD	discard signal	child status has changed
SIGTTIN	stop process	background read attempted
SIGTTOU	stop process	background write attempted
SIGIO	discard signal	I/O is possible on a descriptor
SIGXCPU	terminate process	cpu time limit exceeded
SIGXFSZ	terminate process	file size limit exceeded
SIGVTALRM	terminate process	virtual time alarm
SIGPROF	terminate process	profiling timer alarm
SIGWINCH	discard signal	Window size change
SIGINFO	discard signal	status request from keyboard
SIGUSR1	terminate process	User defined signal 1
SIGUSR2	terminate process	User defined signal 2
SIGWAKE	start process	Wake upon reaching end of long, boring list of signals

A little puzzle

Signals are a kind of interprocess communication

Q: Difference between signals and pipes or shared memory?

A:

- Asynchronous notification
- Doesn't send a "message" as such; just a signal number
- Puzzle: Then how could I do *this*.....*?*

Run demo