LECTURE M4

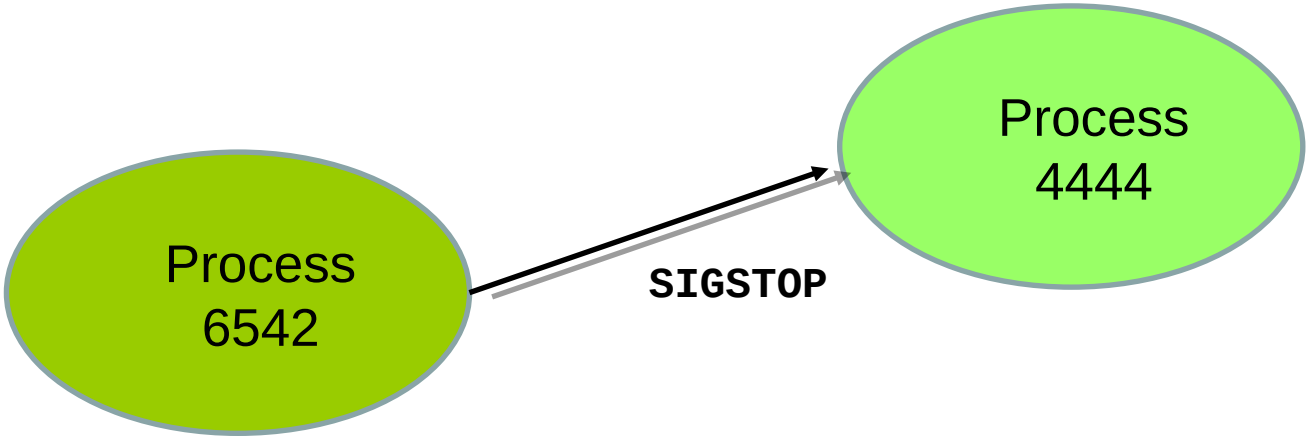
Asynchronous Process Events

Synchronous and Asynchronous

- Synchronous events
  - Events are only notified when asked for
    - E.g. select
- Asynchronous events
  - System notifies when I/O event happens
    - E.g. signals

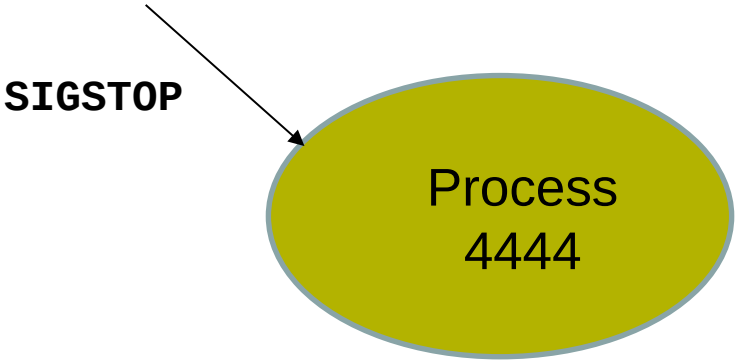
Signals

- Signals are used to notify a program of a particular event
  - ~ software interrupts
  - Signals are asynchronous I/O
  - No data is exchanged!



Signals

- Principle
  - OS or process ‘raises’ a signal to some process
  - The receiving process’ signal handler handles the signal



Signal handler

SIGSTOP	*Ref to handler code*
SIGABRT	*Ref to handler code*
SIGINT	*Ref to handler code*
…	*Ref to handler code*

Signals

Signal constants are defined in signal.h

Signal	Description
SIGABRT	Process abort signal.
SIGALRM	Alarm clock.
SIGFPE	Erroneous arithmetic operation.
SIGHUP	Hangup.
SIGILL	Illegal instruction.
SIGINT	Terminal interrupt signal.
SIGKILL	Kill (cannot be caught or ignored).
SIGPIPE	Write on a pipe with no one to read it.
SIGQUIT	Terminal quit signal.
SIGSEGV	Invalid memory reference.
SIGTERM	Termination signal.
SIGUSR1	User-defined signal 1.
SIGUSR2	User-defined signal 2.
SIGCHLD	Child process terminated or stopped.
SIGCONT	Continue executing, if stopped.
SIGSTOP	Stop executing (cannot be caught or ignored).
SIGTSTP	Terminal stop signal.
SIGTTIN	Background process attempting read.
SIGTTOU	Background process attempting write.
SIGBUS	Bus error.
SIGPOLL	Pollable event.
SIGPROF	Profiling timer expired.
SIGSYS	Bad system call.
SIGTRAP	Trace/breakpoint trap.
SIGURG	High bandwidth data is available at a socket.
SIGVTALRM	Virtual timer expired.
SIGXCPU	CPU time limit exceeded.
SIGXFSZ	File size limit exceeded.

Signals

- The shell command
  - Kill –SIGKILL <pid>
    - Sends ‘SIGKILL’ (signal #9) to process <pid>
- Send a signal to another process
  - int kill(pid, signal_id)
  - Examples
    - Kill(1234, SIGTERM)
      Send the termination signal to the process with id 1234
- Send a signal to ‘yourself’
  - int raise(signal_id)

Signals

- Set your own handler for a signal
  - int sigaction(int signum, const struct sigaction *act,
    struct sigaction *oldact);

```
struct sigaction {
    void     (*sa_handler)(int);
    void     (*sa_sigaction)(int, siginfo_t *, void *);
    sigset_t   sa_mask;
    int        sa_flags;
    void     (*sa_restorer)(void);
    };
```

Signals

- Study the code … (simple example)

[Toledo: Signals]