Why POSIX Signals?

Problems with the `signal()` function:

- cannot determine the current action for a signal

$\leadsto$ may change the action but not restore previous state

- in a signal handler, other signals cannot be blocked

$\leadsto$ signal handler may be interrupted by other signals

- cannot *block* signals (delay delivery until *unblock*)

Data Structure

```
/* simplified from /usr/include/bits/sigaction.h  */

struct sigaction {
   /* pointer to handler function */
   void(*)(int) sa_handler;

   /* signals to be blocked while handler running */
   sigset_t  sa_mask;

   /* flags normally set to 0 */
   int       sa_flags;
   /* if SA_NOCLDSTOP
      => don't send SIGCHLD when children stop. */
}
```

Manipulating a sigset_t

A `sigset_t` contains signals to be blocked.

- `int sigemptyset(sigset_t *set);`
  initialize (no signals to be blocked)
- `int sigfillset(sigset_t *set);`
  initialize (all signals to be blocked)
- `int sigaddset(sigset_t *set, int signum);`
  add signal number `signum` to the set
- `int sigdelset(sigset_t *set, int signum);`
  remove signal number `signum` from the set
- `int sigismember(const sigset_t *set, int signum);`
  return 1: `signum` is in the set
  return 0: `signum` is not in the set
  return -1: `signum` is an invalid signal number

Installing a Signal Handler

```
int sigaction(int signum,
              const  struct  sigaction  *act,
              struct sigaction *oldact);
```

signal handler for `signum` $\rightsquigarrow$pointer `act`.

previous handler $\rightsquigarrow$pointer `oldact`.

Temporarily Block Signals

During critical sections of code, the delivery of signals may be unwanted.

Set these according to a `set` via a call to

```
int  sigprocmask(int  how,  const  sigset_t *set, sigset_t *oldset);
```

The previous list of blocked signals is saved in `oldset`.

The parameter `how` determines the following

how	description
`SIG_SET`	`set` is the new mask of blocked signals
`SIG_BLOCK`	`set` contains signals to be blocked
`SIG_UNBLOCK`	`set` contains signals to be unblocked

Pending Signals

When a signal `signum` is sent while the process has blocked `signum`, the signal is said to be *pending*.

The signal will then be delivered after it is unblocked.

A process may get a mask of currently pending signals by calling

```
int sigpending(sigset_t *set);
```

After the call, the memory pointed to by `set` contains the signals which are blocked and pending.