Signal Handling

Signal Handling Topics

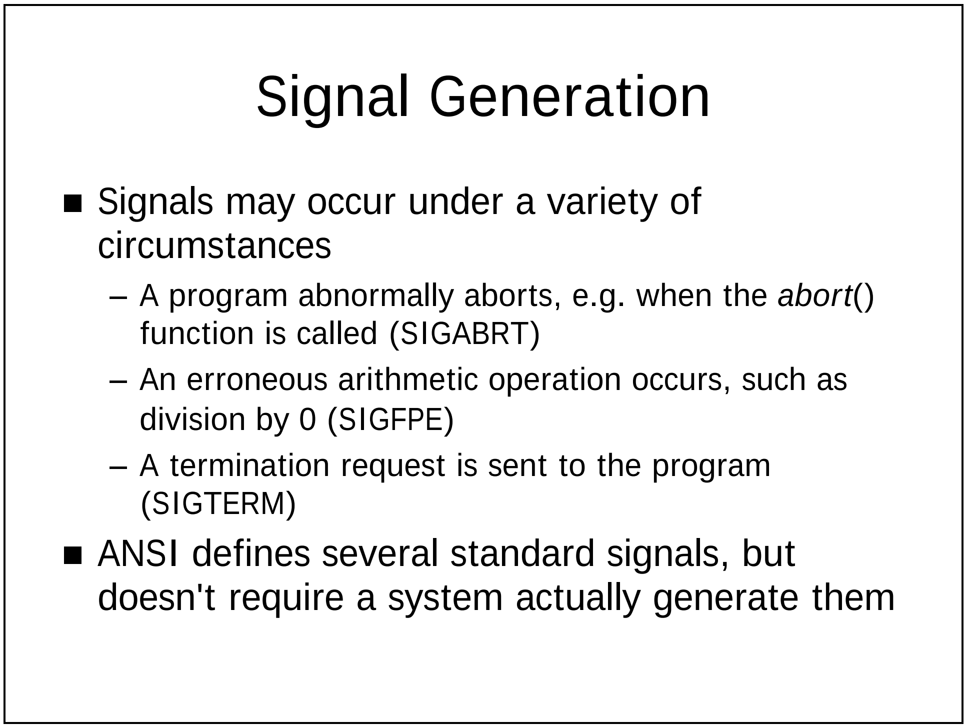
- What are signals?
- Signal states
- Signal Generation
- signal()
- Application-defined Signal Handler
- Related Functions

What are signals?

- Signals are analogous to hardware interrupts
- Signals can be sent to processes from other process, from the OS or from the process itself
- The purpose off the signal is to notify of an event

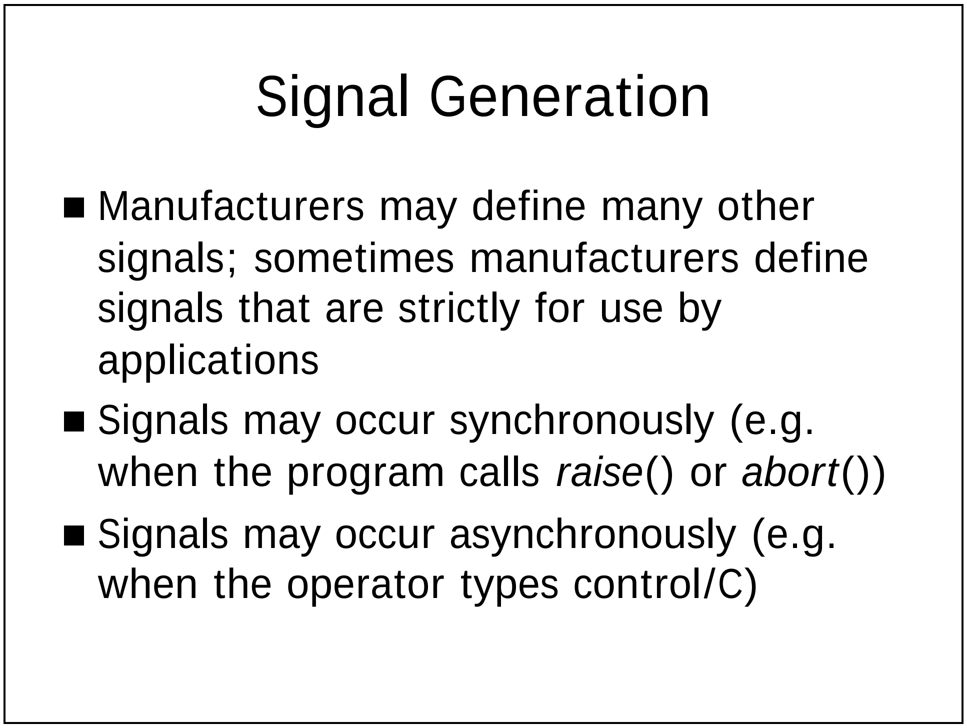
Signal States

- When sent
  - signal posted or generated
- When sent, before received
  - signal pending
- When process receives notification
  - signal delivered



Signal Generation

- Signals may occur under a variety of circumstances
  - A program abnormally aborts, e.g. when the *abort*() function is called (SIGABRT)
  - An erroneous arithmetic operation occurs, such as division by 0 (SIGFPE)
  - A termination request is sent to the program (SIGTERM)
- ANSI defines several standard signals, but doesn't require a system actually generate them



Signal Generation

- Manufacturers may define many other signals; sometimes manufacturers define signals that are strictly for use by applications
- Signals may occur synchronously (e.g. when the program calls *raise*() or *abort*())
- Signals may occur asynchronously (e.g. when the operator types control/C)

signal()

- void (*signal(int sig, void (*handler)(int sig)))(int sig);
  - #include <signal.h>
  - *sig* must be an implementation defined signal
  - *handler* must be:
    - SIG_IGN to tell the system to ignore the signal (note that some signals cannot be ignored)
    - SIG_DFL to tell the system to perform default signal handling
    - An application defined signal handler (more on this later)

signal()

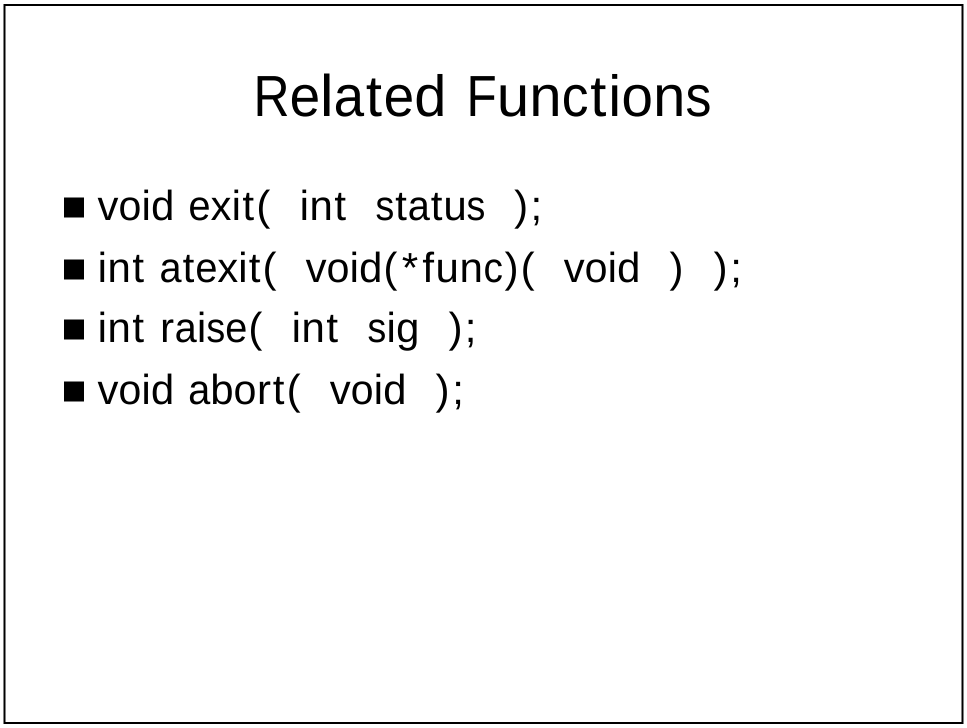
- void (*signal(int sig, void (*handler)(int sig)))(int sig);
  - Return value:
    - SIG_ERR if the operation is rejected
    - The previously registered signal handler (may be the value SIG_DFL)
    - Special note: *SIG_IGN* and *SIG_DFL* are NOT really functions

Application-defined Signal Handler

- Should perform limited processing; note that:
  - Calls to C library functions other than *signal()* result in undefined behavior
  - Execution of *return* may result in undefined behavior
- May exit via three actions:
  - return
  - exit()
  - long jump

Application-defined Signal Handler

- Upon signal detection, the system sets the handler for the detected signal to SIG_DFL...
- ... in order for a signal handler to remain in effect it must be reregistered following signal detection; handlers may do this internally
- Example
  - http://faculty.washington.edu/sproedp/advc/csamples/less20.c.html



Related Functions

- void exit(int status);
- int atexit(void(*func)(void));
- int raise(int sig);
- void abort(void);