Introduction

- A **signal** is a mechanism for notifying a process that an event has occurred.
  - When a signal is sent to a process its normal execution is interrupted

- Events can arise from executing an instruction in the process's instruction stream
  - Illegal instruction e.g., divide by zero
Introduction

- Events occur at any time and come from an external source
  - may be unrelated to the execution of the process
  - e.g., ctrl-C

- Upon receipt of a signal a process may take some action
  - Take a default action; or
  - Use a pre-defined signal handler
Signal Handling

- The system call `signal()` captures a specific event and associates it with a programmer-defined function.

- To use the `signal` system call requires that you include `signal.h`.
Example

```c
int alarmflag=0;
alarmHandler ()
{
    printf("An alarm clock signal was received\n");
    alarmflag = 1;
}
main()
{
    signal (SIGALRM, alarmHandler);
    alarm(3);
    printf("Alarm has been set\n");
    while (alarmflag==0);
    printf("Back from alarmHandler function\n");
}
```

Sets up signal handler

Instructs OS to send SIGALRM in 3 seconds
Important Signals

- **SIGINT**
  - Interrupt signal from terminal (ctrl-c)
- **SIGALRM**
  - Alarm signal from OS