CS 153 Lab3

Kishore Kumar Pusukuri

Outline

wait and waitpid
signals

- When a process terminates, either normally or abnormally, the parent is notified by the kernel sending the parent the SIGCHLD signal.
- The termination of a child is an asynchronous event, so this signal is the asynchronous notification from the kernel to parent.
- The parent can choose to ignore this signal, or it can provide a function that is called when the signal occurs (a signal handler).
- The default action for this signal is to be ignored.

cont...

A process that calls wait or waitpid can :

- block (if all of its children still running), or
- return immediately with the termination status of a child (if a child has terminated and is waiting for its termination status to be fetched), or
- return immediately with an error (if it doesn't have any child process).

```
#include <sys/types.h>
#include <sys/wait.h>

pid_t wait (int *statloc);

pid_t waitpid (pid_t pid, int *statloc, int options);
```

Both return: process ID if OK, 0, or -1 on error.

For both functions the argument ‘‘statloc’’ is a pointer to an integer. If this argument is not a null pointer, the termination status of the terminated process is stored in the location pointed to by the argument.

wait vs waitpid

The differences between these two function are :

- wait can block the caller until a child process terminates, while waitpid has an option that prevents it from blocking.
- If the caller has multiple children, wait returns when one terminates. waitpid has a number of options that control which process it waits for.

signals

We will study the programs....


Thaaaank
Yoooouuu!!!