Semaphore (6A)

- Semaphore

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Young Won Lim
12/06/2012

Semaphore

```
#include <signal.h>

typedef void (*sighandler_t)(int);

void myfunc1(int);
void myfunc2(int);


sighandler_t signal(int signum, sighandler_t handler);


signal(SIGINT, myfunc1);
func = signal(SIGINT, myfunc2);
```

Signal Example (1)

```
int main (void)
{
   int i;
   void (*fn)(int);

  signal(SIGINT, myfun1);

  for (i=0; i<10; i++) {
    printf("Hello world. i=%d\n", i);  sleep(1);
  }

  fn = signal(SIGINT, SIG_DFL);

  printf("in main:  calling fn = signal(SIGINT, SIG_DFL) ... \n");
  printf("in main:  fn points to myfun1 ... \n");
  printf("in main:  thus, fn(100) ==> myfun1(100) is called again ... \n");

  printf("myfun1 =%p, fn =%p \n", (void *) myfun1, (void *) fn);

  fn(100);

  i=0;
  while (1) {
    printf("Hello world. i=%d \n", i++);  sleep(1);
  }

  return 0;
}
```

```
void myfun1(int signo)
{
   printf("in myfunc1()... signo=%d \n", signo);
}
```

Signal Example (2)

```
Hello world. i=0
Hello world. i=1
^Cin myfunc1()... signo=2
Hello world. i=2
Hello world. i=3
Hello world. i=4
^Cin myfunc1()... signo=2
Hello world. i=5
Hello world. i=6
Hello world. i=7
^Cin myfunc1()... signo=2
Hello world. i=8
Hello world. i=9
in main:  calling fn = signal(SIGINT, SIG_DFL) ...
in main:  fn points to myfun1 ...
in main:  thus, fn(100) ==> myfun1(100) is called again ...
myfun1 =0x8048474, fn =0x8048474
in myfunc1()... signo=100
Hello world. i=0
Hello world. i=1
Hello world. i=2
^C
```

NULL Signal

int kill(pid_t pid, int sig);

send a signal to a process or a group of processes specified by pid.
The signal to be sent is specified by sig in the list given in <signal.h> or 0
If sig is 0 (the null signal), error checking is performed but no signal is actually sent.
The null signal can be used to check the validity of pid.

An implementation that provides extended security controls may impose further implementation-defined restrictions on the sending of signals, including the null signal. In particular, the system may deny the existence of some or all of the processes specified by pid.

EINVAL
The value of the sig argument is an invalid or unsupported signal number.
EPERM
The process does not have permission to send the signal to any receiving process.
ESRCH
No process or process group can be found corresponding to that specified by pid.

Reference

References

[1] http://en.wikipedia.org/
[2] http://www.tldp.org/LDP/lpg/node46.html