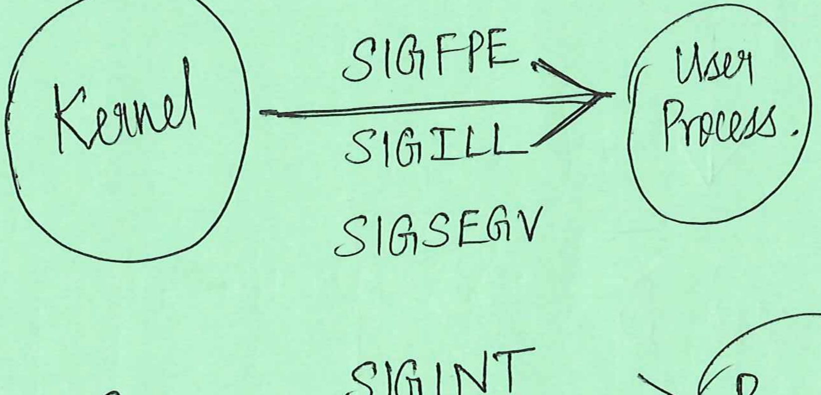
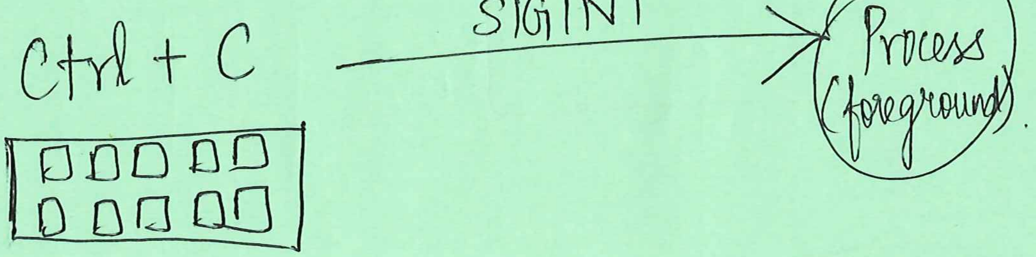
Signals ①

Unix signal - higher level s/w form of ECF.
- allows processes and kernel to interrupt other processes.

Signal - small message that notifies a process than an event of some type has occurred in the system.

eg. SIGINT	-	interrupt from keyboard.
SIGFPE	-	Floating point exception.
SIGKILL	-	Kill Program
SIGSEGV	-	Invalid memory reference
SIGILL	-	Illegal Instruction.





Signal Terminology

1. Sending a signal.

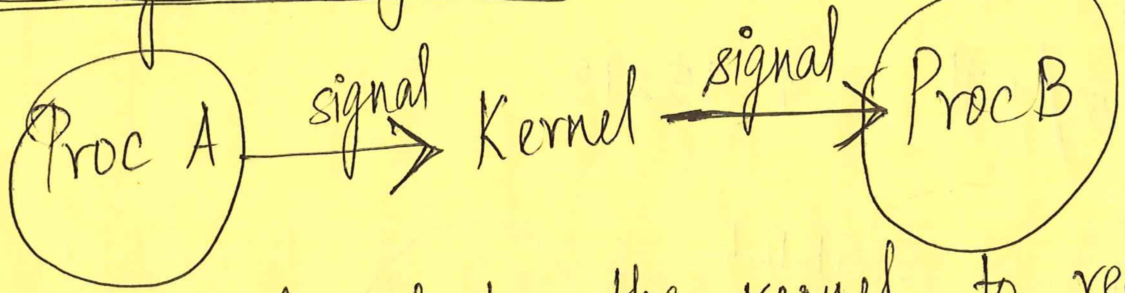
$$\boxed{\text{Kernel} \longrightarrow \text{Process}}$$

Reasons for sending a signal:

i) Kernal has detected a system event
eg. ÷ by 0
NULL pointer dereference.

ii)
$$\begin{tikzcd}
\boxed{\begin{array}{l}\text{Process A}\\ \text{kill(pid, signal)}\end{array}} \arrow[d] \\
\text{Kernel} \arrow[d] \\
\boxed{\text{Process B.}}
\end{tikzcd}$$

2. Receiving a signal



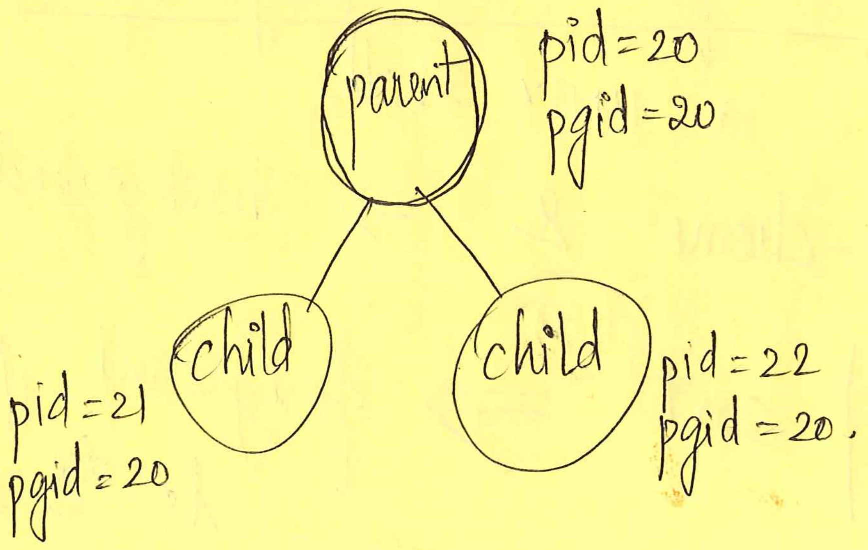
Proc B is forced by the kernel to react to the signal in some way.

Proc B can either

1) Ignore the signal
2) Terminate
3) Catch the signal by executing a user-level function. i.e. "signal handler"

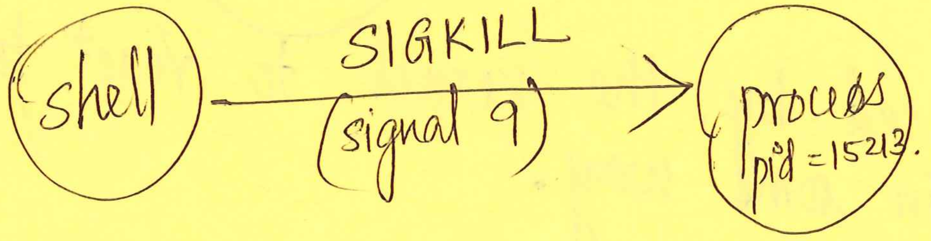
Process Groups

Each process belongs to exactly one process group.

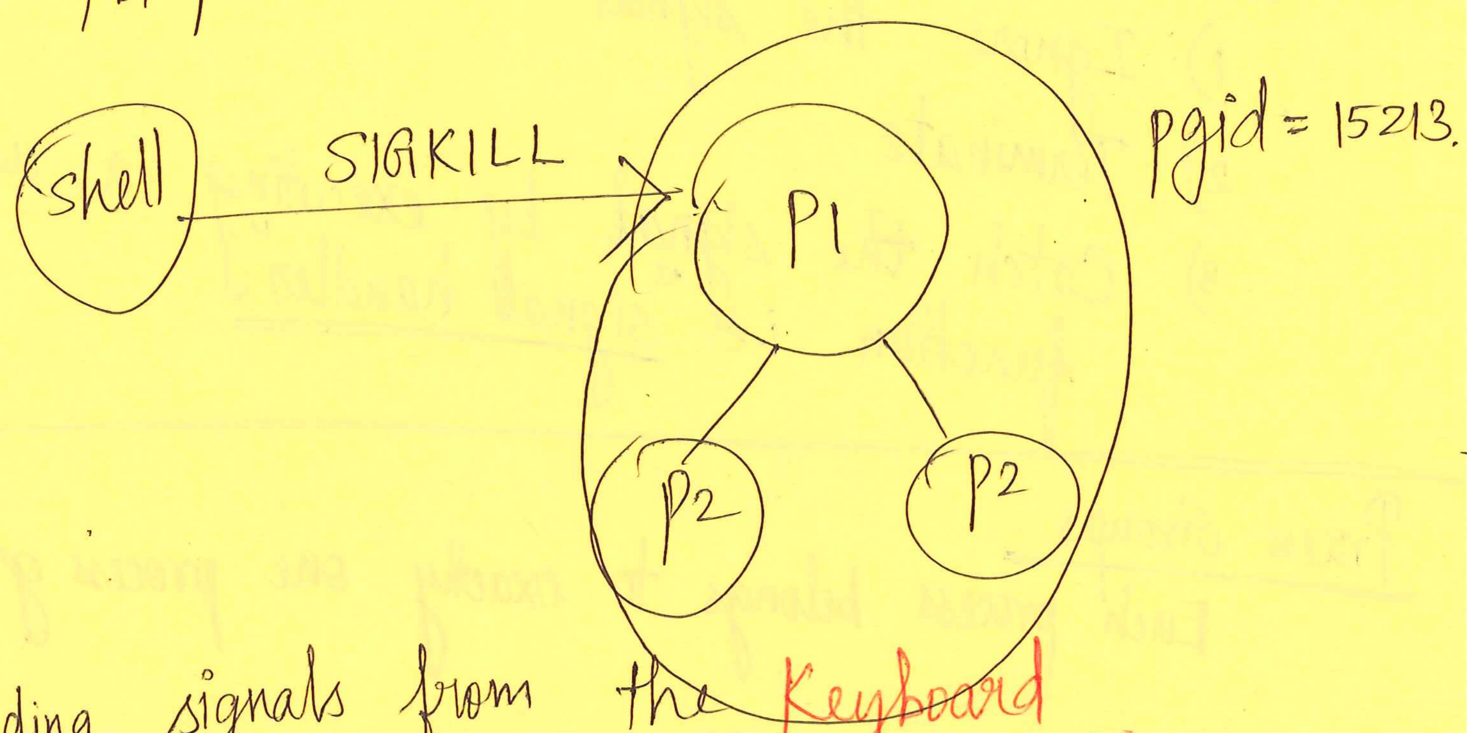


1. Sending signals with /bin/kill program

```
$ /bin/kill -9 15213
```



```
$ /bin/kill -9 -15213
```



2. Sending signals from the Keyboard

"Job" in a UNIX shell.

$ google-chrome & ⟹ background job.

$ ls | sort ⟹ foreground job.

$ls \xrightarrow{pipe} sort \rightarrow o/p$

8 $$\begin{tikzcd} \text{Ctrl+C} \arrow[r, "\text{SIGINT}"'] & \text{shell} \arrow[r, "\text{SIGINT}"'] & \text{Foreground job / process group} \end{tikzcd}$$ (FG)

Only one FG job at a time.

Default action for SIGINT ⇒ Terminate the (process) foreground job

$$\begin{tikzcd} \text{CTRL+Z} \arrow[r, "\text{SIGTSTP}"'] & \text{shell} \arrow[r] & \text{Foreground process group} \end{tikzcd}$$

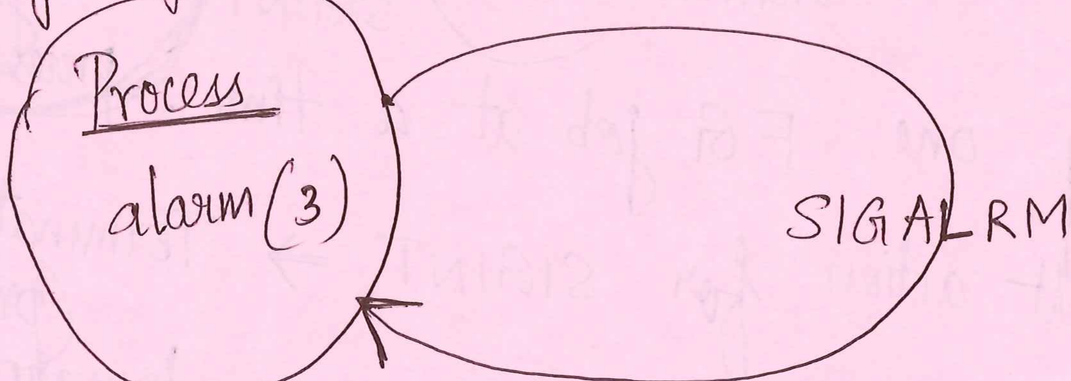
DEFAULT ACTION: STOP (Suspend) the foreground job.

3. Sending signals with the "kill" function

```
#include <sys/types.h>
#include <signal.h>
int kill(pid_t pid, int sig);
```

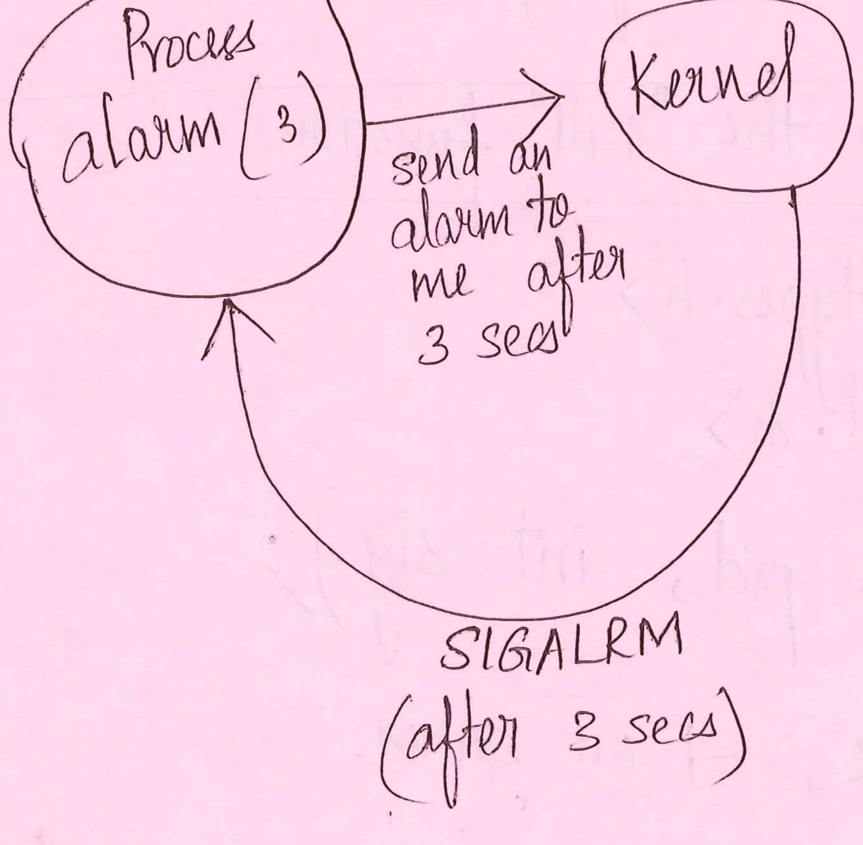
Returns: 0 if OK, −1 on error.

4. Sending signals with the "alarm" function.



```
#include <unistd.h>
unsigned int alarm(unsigned int secs);
```

Returns: Remaining secs of previous alarm, or 0 if no previous alarm.



alarm(2);
⇓
cancels any pending alarms