I2180
LINUX :
APPELS SYSTÈME

LES SIGNAUX
(SUITE)

alarm

```
#include <unistd.h>

unsigned int alarm (unsigned int seconds);
```

- Où: seconds : durée de temporisation avant émission d'un signal SIGALRM au processus appelant
- Il n'y a qu'une temporisation *alarm* par processus
- Si *seconds* > 0 : définit une nouvelle temporisation (en annulant éventuellement la précédente)
- Si *seconds* = 0 : annule la temporisation en cours
- Renvoie le nombre de secondes restantes de l'alarme précédemment programmée ; 0 si aucune alarme n'était en cours

POSIX

Jeu de signaux

```
#include <signal.h>

int sigemptyset (sigset_t* ensemble);
int sigfillset (sigset_t* ensemble);
int sigdelset (sigset_t* ensemble, int sig);
int sigaddset (sigset_t* ensemble, int sig);
int sigismember (const sigset_t* ensemble, int sig);
```

- Fonctions permettant respectivement de vider ou remplir un jeu de signaux, supprimer ou ajouter un signal à un jeu de signaux, tester l'appartenance d'un signal à un jeu de signaux.
- Où: ensemble : jeu de signaux

  sig : numéro de signal ∈ [1,NSIG]
- Renvoie 0 si réussi ; -1 si échec (à l'exception de *sigismember*)

➔ Pour manipuler des masques de signaux (ex: sa_mask de sigaction)

POSIX

sigprocmask

```
#include <signal.h>
int sigprocmask (int how,
                 const sigset_t* set,
                 sigset_t* oldset);
```

- Modification du masque de signal courant pour bloquer/débloquer des signaux
- Où: how : modification de l'action du signal (SIG_BLOCK, SIG_UNBLOCK, SIG_SETMASK)

  set : nouveau jeu de signaux (bit=1 $\Rightarrow$ signal à modifier)

  oldset : sauvegarde de l'ancien masque
- Renvoie 0 si réussi ; -1 si échec
- Lors d'un *fork*, le processus fils reçoit le même masque de signaux que son père.

EXEMPLE

- Cf. exemple2.c

- Compilation avec l'option:

  `-D_POSIX_C_SOURCE`