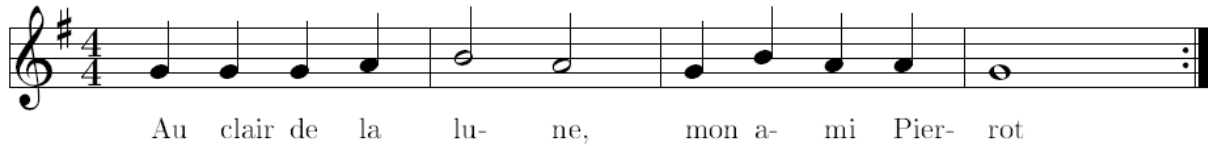


TD n° 4b

Pilotes et Périphériques Matériels

Le but de ce TD est d'ajouter un module Karaoké à notre noyau ! Même si celui-ci est limité (nous serons limités aux partitions utilisant seulement 3 notes !), il devra afficher cette partition en utilisant les LEDs du clavier.



Vous pouvez utiliser le script se trouvant à l'adresse suivante pour vous éviter de retaper la partition :

<http://kistren.polytech.unice.fr/cours/sae/td4b/music.sh>

Vous disposez de 4 heures pour réaliser ce TD qui sera noté. A vos claviers !

TD n° 4b

Pilotes et Périphériques Matériels

1 Introduction

Pour la mise en place de ce TD, nous vous invitons à réaliser les tests dans votre machine virtuelle pour éviter un plantage de votre machine physique en cas de mauvaise utilisation des adresses de fonctions que vous utiliserez. Vous pourrez ensuite tester sur la machine physique pour réellement faire allumer les LEDs du clavier.

1.1 Point de départ

Pour trouver l'état des LEDs, utiliser les déclarations suivantes :

```
#include <linux/module.h> /* Necessary for the definition of a module */
#include <linux/init.h> /* Necessary for the macros __init and __exit */

#include <linux/kernel.h>
#include <linux/fs.h> /* file_operations */
#include <asm/uaccess.h> /* get_user() and put_user() */

#include <linux/kbd_kern.h> /* kbd_struct, kbd_table */
#include <linux/vt_kern.h> /* fg_console */
```

1.2 Les fonctions et variables utiles

```
/* Pointeur sur les fonctions et informations utiles du noyau */
unsigned char (* my_getledstate) (void);
void (* my_setledstate) (struct kbd_struct *kbd, unsigned int led);
struct kbd_struct *my_kbd_table;
```

Seule la variable `kbd_table` est exportée dans le noyau et vous permettra d'initialiser votre propre variable `my_kbd_table`. Les variables du module devront être initialisées aux adresses des variables du noyau : `getledstate`, `setledstate`. L'initialisation de ces variables se fera lors du chargement du module. Par exemple, l'appel :

```
> insmod led.ko my_getledstate=0x1234 ...
```

permet de charger le fichier `led.o` et d'initialiser la variable `my_getledstate` à l'adresse `0x1234`.

Les adresses des 2 variables nécessaires aux fonctions du module pourront être trouvées dans le fichier `/boot/System.map-2.6.x` construit lors de la compilation du noyau.

Écrivez un script shell pour charger le module et lui passer les bonnes adresses pour ces variables.

2 Information sur l'accès aux LEDs du clavier

En ce qui concerne les LEDs, elles peuvent être allumées/éteintes en mettant à 0/1 un bit suivant les conventions :

- Scroll Lock : bit 0
- Num Lock : bit 1
- Caps Lock : bit 2

Ainsi, si les LEDs « Scroll Lock » et « Caps Lock » sont allumées, la valeur retournée par la fonction `getledstate` sera 5 (4 + 1). L'écriture d'une LED se fait de la façon suivante :

```
my_setledstate(my_kbd_table + fg_console, valeur);
```

TD n° 4b

Pilotes et Périériques Matériels

2.1 Lecture sur le fichier spécial

Afin de vérifier l'état des LEDs, on permet la lecture du fichier `/dev/led`. Un exemple d'utilisation de ce fichier est donné ci-dessous :

```
> cat /dev/led
Etat des LEDES:
Scroll:on
Caps:off
Lock:off
```

Vous utiliserez bien entendu la fonction `my_getledstate` (pointeur sur la « vraie » fonction `getledstate` du noyau).

2.2 Ecriture sur le fichier spécial

Créez le fichier spécial `/dev/led` de type caractère de numéro de majeur 100.

L'écriture sur ce fichier spécial permettra de changer l'état des LEDs de la console virtuelle courante. Chaque caractère écrit sur ce fichier est considéré comme une commande avec les conventions suivantes :

- n : éteindre num-lock
- N : allumer num-lock
- s : éteindre scroll-lock
- S : allumer scroll-lock
- c : éteindre caps-lock
- C : allumer caps-lock
- o : éteindre toutes les LEDs
- 1 : allumer toutes les LEDs
- t : délai de 200 millisecondes

Les autres caractères sont ignorés.

Afin de vérifier que votre programme fonctionne correctement, vous pouvez essayer de l'exécuter avec le script shell suivant :

```
while true; do echo "CtctCtcttttttt" > /dev/led; done
```

Bien entendu, pour exécuter ce script et faire allumer les LEDs du clavier, vous testerez sur votre machine physique, l'émulateur ne permettant pas un accès direct au matériel. Vous pourrez aussi utiliser le script se trouvant à l'adresse suivante, celui-ci contenant une partition plus longue :

<http://kistren.polytech.unice.fr/cours/sae/td4b/music.sh>

3 IOCTL

On décide de pouvoir éventuellement rendre bloquant l'accès aux LEDs (i.e. en mode bloquant, on ne peut pas lire ou écrire sur le fichier `/dev/led` s'il est utilisé par un autre processus). La mise en place du mode bloquant/non bloquant se fera par l'utilisation de la primitive `ioctl` avec les constantes suivantes :

- `LED_SETBLOCK` (définie à 1, un paramètre) permet de mettre/enlever le mode bloquant
- `LED_GETBLOCK` (définie à 2, un paramètre) permet de stocker à l'adresse passée en paramètre la valeur de la variable du noyau contenant le mode d'accès aux LEDs.

Vous pourrez trouver sur le site un fichier de test pour les valeurs modifiables par `ioctl` :

<http://kistren.polytech.unice.fr/cours/sae/td4b/ioctl.c>