ASE - Contrôle TD (durée 1h30 – documents autorisés)

Exercice 1 : Questions

Question 1:

Que stockent les registres `eip`, `esp` et `ebp` ? Faites un (ou plusieurs) schéma(s) pour expliquer le contenu de ces registres lors de l'appel d'une fonction f, puis lorsque f se termine.

Question 2:

Dans l'ordonnancement préhemptif, pour quelle raison ajoutons-nous autour de certaines parties de code des appels à `irq_disable()` et `irq_enable()` ? Donnez un exemple critique dans lequel ces appels sont indispensables et expliquez pourquoi le programme pourrait crasher sans eux.

Question 3:

Multitâche coopératif et multitâche préhemptif : expliquez les deux principes en soulignant les différences, et commentez.

Question 4:

Retour au `main` : expliquez en le principe avec un schéma, puis proposez et commentez votre solution (en précisant "De mémoire" si vous ne l'avez pas sous les yeux).

Exercice 2 : Mécanisme de signaux

d'après un examen d'ASE de 2004

On souhaite ajouter un mécanisme simple de signaux entre contextes d'exécution à notre ordonnanceur. Tous contexte d'exécution peut demander la transmission d'un signal à un autre contexte donné, à l'aide de la fonction :

```
void sendSignal(ctx_s * ctx, void * argfh);
```

De plus chaque contexte peut (éventuellement) associer un *signal handler* `void fh(void *)` pour tout signal reçu à l'aide de la fonction

```
void setSignalHandler(sig_t fh);
```

En cas de réception du signal, ce signal handler, s'il a été associé, est exécuté (avec l'argument `argfh`) **avant** que le contexte visé ne soit normalement réactivé. Si aucun *signal handler* n'est positionné, alors le signal est perdu.

Question 1: Structure `ctx_s`

Les contextes d'exécution doivent connaître un nouvel état nommé `CATCH_SIGNAL`. Cet état doit être positionné lorsque l'on demande le déclenchement d'un signal sur un contexte. De plus, la structure de données `ctx_s` doit être modifiée afin de prendre l'adresse de la fonction *signal handler*, ainsi que son argument `argfh`.

Proposez une modification du fichier `ctx.h`

Question 2: setSignalHandler

Proposez une implantation de cette fonction. Elle écrasera éventuellement un précédent *signal handler* enregistré.

Question 3: sendSignal

Proposez une implantation de cette fonction.

Question 4: switchToCtx

Expliquez pourquoi il n'est pas possible d'exécuter le *signal handler* **après** avoir restauré les registres `esp` et `ebp` de ce contexte. Donnez une nouvelle implantation de `switchToCtx` qui exécute un éventuel *signal handler* associé au contexte à réactiver.

Question 5: switchToCtx

Expliquez, dans le cadre d'un multitâche (i) préhemptif, (ii) coopératif, dans quelle pile ce *signal handler* est exécuté, et commentez.