

Annexe E

Travaux Pratiques 3

L'objectif de ce tp est de mettre en pratique les concepts présentés dans le chapitre 8 à savoir, les interruptions matérielles.

Nous vous demandons de télécharger depuis le serveur SVN (File->import->other->serveur svn etc... (voir chapitre5.) le projet TP3_Eleves et de nommer l'importation TP3 (dernier écran d'importation).

Informations :

Pour un mapping correct entre votre clavier et qemu, nous vous conseillons de rajouter l'option « -k fr » (pour clavier français) à la ligne de commande de qemu.

Question "sur le TP n-1" - Durée estimée : 30 minutes

Pour cette question, rajouter aux options de `qemu` : `-soundhw pcspk -m 5`

Rajouter dans votre main au bon endroit les lignes suivantes (et les inclusions aux fichiers nécessaires, déclaration de variables etc ...) :

```
1 while(1) {  
2     son.note(clavier.getchar());  
3 }
```

Listing E.1 – Dans main.cpp

Puis compléter la classe `Son.cpp` du répertoire `drivers` pour vous permettre de faire bipper le PC. Il faut rajouter une ligne dans la méthode `void Son::jouer(int nFrequence)`

Question 1 - Durée estimée : 60 minutes

Développez une application Horloge qui à partir de la méthode `getSeconde()` du pilote de périphérique `timer` affiche les minutes et secondes au milieu du mur du haut (voir la figure 8.4 du chapitre 8). Il s'agit de compléter le squelette dans `Applications/Horloge.cpp`.

Pour cela utilisez les méthodes `afficherNombre(...)` et `afficherMot(...)` de la classe `Ecran`.

Question 2 - Durée estimée : 60 minutes

Développez la gestion de la raquette dans `Applications/SuperPong/Grille.cpp` et `SuperPong.cpp`.

Dans un premier temps, étudiez les sources de l'Application SuperPong. Ensuite modifier le fichier `Application/SuperPong/SuperPong.cpp` puis le fichier `Grille.cpp`.

Dans le premier fichier, cela consiste à tester les valeurs des caractères entrés au clavier puis d'appeler correctement les méthodes `DescendreRaquette(in)` et `monterRaquette(int)`

Pour `Application/SuperPong/Grille.cpp` vous devez modifier la valeur des objets `raqd` et `raqg` via les méthodes appropriées. La raquette est vue comme un petit mur partant de `getHaut()` jusqu'à `getBas()` sur la colonne `getY()`, méthodes de la classe `Raquette`. Il suffit, via ses méthodes, d'affecter les cases écrans aux bonnes valeurs (via `setCaseBord()` ou `setCaseVide()`).

Question 3 - Durée estimée : 60 minutes

Développez la méthode `avance()` de la classe `balle` dans `Applications/SuperPong/Balle.cpp`.

Ici la balle va se déplacer sur l'écran. Une balle en (x,y) se déplace en $(x+dx,y+dy)$ ou dx et dy prennent les valeurs 1 ou -1. Lorsque la future position est un mur, la balle rebondie. Dans ce cas, une des valeurs dx,dy , voir les deux, inverse leur valeur.

Si la balle sort de l'écran (cas du trou derrière les raquettes), la balle est perdue (pour l'instant elle se limite à ne plus avancer)

Question pour la prochaine fois - Durée estimée 60 minutes

Développez un handler spécifique pour notre jeu (Il s'agit pour le handler d'appeler directement les méthodes `monter` et `descendre` de la classe `Grille`)

Lorsqu'une balle est lancée, seule la balle fonctionne, pourquoi ?. Lorsque nous utilisons le handler de la question 4, nous pouvons de nouveau bouger les raquettes. Pourquoi ?

La prochaine fois : la notion d'activité puis de multi-activités.