TP2 - MiniKernel avec Gestion des interruptions et du clavier
Programmation Système Avancée

Rudolf Höhn & Axel Fahy

November 24, 2015

1. **Comment vous êtes-vous réparti le travail ?**
   Nous avons travaillé en parallèle et à chaque fois, les deux ensemble.
2. **Votre kernel comporte-il des bugs ? Si oui lesquels et comment pourriez-vous les corriger ?**
   Non.
3. **Dans quel ordre vous avez initialisé les différents points ci-dessus dans votre kernel ? Justifiez**
   On initialise dans cet ordre là :
   (a) GDT
   (b) Screen
   (c) PIC
   (d) "sti" => Réactivation des interruptions
   (e) IDT
   (f) Timer
   (g) Keyboard

   Nous avons choisi cet ordre car pour le screen et la GDT, ils faisaient partis du TP1. Puis, entre le PIC et le Keyboard, c'est un ordre de besoin. Par exemple, l'IDT ne peut pas être initialisée sans avoir réactiver les interruptions.
4. **Pourquoi remappe-t-on les IRQ 0 à 7 aux interruptions 32 à 39 ?**
   Pour ne plus que les IRQ entrent en conflit avec les exceptions processeurs (qui sont des interruptions allant de 0 à 31).
5. **Que se passerait-il si on ne le faisait pas ?**
   On ne pourrait pas faire la différence entre par exemple une interruption de timer et une exception de division par 0.
6. **Comment pouvez-vous tester que votre gestionnaire d'interruption pour les exceptions fonctionne correctement ?**
   On ne peut pas tous les tester, mais on a reussi à en tester certaines.
7. **Quelles exceptions avez-vous pu générer et comment avez-vous fait ?**
   En faisant un 'return' dans le kernel lorsqu'on reçoit un 'Q' pour quitter l'OS, et que dans le 'bootloader.s' on ne rentre pas dans une boucle infinie, on reçoit l'exception :
   Exception 6 - Invalid Opcode (Undefined Opcode)

   La seconde exception est celle de la division par zéro. En voulant faire 5/0, on reçoit l'exception :
   Exception 0 - Divide Error
8. **Quelle taille de buffer clavier avez-vous choisie et pourquoi ?**
   La taille de notre buffer est de 1024. Nous avons choisi une taille puissance de 2 pour pouvoir faire un modulo de manière efficace grâce à un ET logique.
9. **Comment pouvez-vous causer une situation de buffer plein quelle que soit la taille du buffer (dans les limites du raisonnable) ?**
   On peut avoir un *sleep* de quelques secondes et dans cette intervalle taper plusieurs caractères. Si le buffer fonctionne, quand on atteint sa taille, on obtient un message d'erreur et dans le cas où le *sleep* se termine sans avoir rempli le timer, les caractères sont réaffichés à l'écran.