Projet libre – Ecriture d'un kernel « DOS-like » en C / ASM

Kernel 32 bits x86

Membres: duc_l -- chef de groupe

perely_k

wakkac_m

Objectifs finaux:

- Bootloader
- Mode protégé
- Filesystem (FAT12)
- Console 80x25
- Clavier
- Souris
- Shell
- Exécutables (système mono tâche)

Déroulement du projet :

Voici les différentes fonctionnalités que nous souhaitons incorporer au kernel. Elles sont listées par ordre chronologique de réalisation.

Le bootloader : *Durée de réalisation : 1 mois*

Le bootloader sera chargé à partir d'une disquette formatée en FAT12. Il sera composé de deux parties (stage1 et stage2). La première sera chargée en mémoire par le bios, la seconde le sera par la première. Il sera bien entendu écrit en assembleur.

Le mode protégé : *Durée de réalisation : 1 mois (réécritures fréquentes)*

Le mode protégé permet d'accéder à un système 32bits, le mode réel étant 16bits. Il s'agit de la partie la plus difficile, car elle nécessite de recoder absolument tout « from scratch », des microcontrôleurs aux interruptions système en passant par les drivers de clavier, écran, souris, têtes de lecture des disques, etc. Elle nécessite la mise en place de la GDT et de l'IDT.

Filesystem : *Durée de réalisation : < 1 mois*

Le FAT12 est le système de fichiers le plus simple, mais aussi le plus largement supporté. Il nous faudra recoder un support pour celui-ci, vraisemblablement pour disquette.

Support hardware : *Durée de réalisation : < 1 mois*

L’usage du clavier et d’une souris devra être supporté, ainsi que l’affichage à l’écran, en mode texte (console). Une invite de commande permettra les interactions entre utilisateur et machine.

Exécutables : *Durée de réalisation : 1 mois*

Le système sera capable d’exécuter des programmes compilés au format ELF.

Détails

Finalité du projet:

Le but de ce projet est purement pédagogique. Il s'agit en effet de comprendre, via sa reprogrammation, le fonctionnement interne d'un kernel très simpliste.

Parmi les thèmes que nous désirons aborder en priorité, et qui constituent les difficultés principales du projet, on peut citer:

- le boot loader.

- la gestion de la memoire.

- recode des appels systemes.

- gestion du filesystem, des drivers de base.

- gestion des processus.

- séparation des privilèges (GDT, ...).

Par ailleurs, il est important de mentionner que les connaissances acquises au cours de l'élaboration de ce type de projet sont, pour ainsi dire, "impérissables".

En effet, chaque nouvelle technologie a besoin d'une base software, une première couche d'abstraction entre le matériel et le développement d'applications.

Moyens techniques:

La province ne disposant pas de vieilles machines physiques pouvant être mises à disposition des étudiants pour de tels projets, nous allons donc utiliser diverses machines virtuelles (Bochs, Qemu, VMWare, …), capables d'émuler différents types de processeurs et périphériques.

Le principal avantage de ces programmes est qu'ils rendent moins fastidieux les phases de tests et de debug. Ils présentent cependant le risque de ne pas émuler parfaitement

le reste des périphériques d'un ordinateur : un kernel qui fonctionne avec Bochs, par exemple, ne fonctionnera pas nécessairement sur une machine réelle. Nous allons donc régulièrement procéder à des tests sur du véritable matériel, appartenant à l'un d'entre nous.

Les langages utilisés seront le C et l'assembleur Intel, assemblé avec NASM et Gas. Les .c seront compilés avec GCC. Le linkage se fera à l'aide du linkeur LD.

Pour la gestion de projet, un serveur SVN a été mis en place, ce qui permet à chaque membre d'avoir accès aux dernières modifications effectuées.

Chronologie du projet:

La première phase consistera a reprogrammer un bootloader en assembleur, qui supportera dans un premier temps les disquettes, puis les disques durs.

Il s'agit là de la partie la plus difficile du projet, qui implique le recode des différentes routines primaires nécessaires au fonctionnement du système, ainsi que des drivers, notamment ceux du clavier, de l'écran et du disque dur.

Une fois cela fait, il nous faudra alors entamer la seconde difficulté du projet, la gestion de la mémoire et des processus, avec le support de formats d'exécutables et de RTL (Run Time Library), comme l'ELF, le a.out, ou COM.

Enfin, nous pourrons procéder au portage de la "toolchain" GNU (binutils, gcc, ...) et disposer ainsi d'un système autonome.