RustOS
Système d'exploitation en Rust

Orphée Antoniadis

Projet de Bachelor - Prof. Florent Glück

Hepia ITI 3ème année

Semestre de Printemps 2017-2018

h e p i a
Haute école du paysage, d'ingénierie
et d'architecture de Genève

Hes·SO GENÈVE
Haute Ecole Spécialisée
de Suisse occidentale

Résumé

Le but de ce projet est d'étudier le langage Rust, en particulier son utilisation pour l'implémentation d'un système d'exploitation de type *bare metal*. Le langage Rust se révèle particulièrement intéressant en tant que digne successeur de C : beaucoup plus robuste que ce dernier et potentiellement tout aussi rapide. La première partie du projet sera de comprendre les paradigmes de programmation utilisés par Rust ainsi que ses caractéristiques principales. Dans un deuxième temps, il s'agira d'implémenter un système d'exploitation très simple, similaire à celui réalisé au cours logiciel « Programmation système avancée » mais écrit en Rust plutôt qu'en C.

Table des matières

1 Introduction **8**
1.1 Contexte . . . 8
1.2 Objectif . . . 8

2 Analyse **8**

3 Conception **8**
3.1 Environnement de développement . . . 8
3.2 Technologies . . . 8
3.3 Architecture . . . 9

4 Rust **9**

5 Système d'exploitation de type *bare metal* **9**
5.1 Compilation . . . 9
5.2 *Linking* . . . 9
5.3 Processus de *boot* . . . 9
5.4 Adressage mémoire . . . 9
5.5 Ports . . . 9
5.6 Interruptions . . . 9
5.7 Periphériques . . . 9
5.7.1 VGA . . . 9
5.7.2 *Timer* . . . 9
5.7.3 Clavier . . . 9

6 Système de fichiers **9**
6.1 Introduction . . . 9
6.2 Structure . . . 9

7 Résultats **9**

8 Discussions **9**
8.1 Problèmes rencontrés . . . 9
8.2 Améliorations possibles . . . 9

9 Conclusion **9**

10 Références **10**

Table des figures

Remerciements

Convention typographique

Lors de la rédaction de ce document, les conventions typographique ci-dessous ont été adoptées.

- Tous les mots empruntés à la langue anglaise ont été écrits en *italique*
- Toute référence à un nom de fichier (ou dossier), un chemin d'accès, une utilisation de paramètre, variable, ou commande utilisable par l'utilisateur, est écrite avec la police d'écriture `Courier New`.
- Tout extrait de fichier ou de code est écrit selon le format suivant :

```
1    fn main() {
2        println!("Hello, world!");
3    }
```

Acronymes

1 Introduction

1.1 Contexte

1.2 Objectif

2 Analyse

3 Conception

3.1 Environnement de développement

La machine utilisée pour le développement du projet est un MacBook Pro avec un processeur Intel à 3 GHz. Il a quand même fallut utiliser une machine virtuelle (VMware) utilisant Linux (Ubuntu 16.04.4 LTS) pour la compilation. Ce choix a été fait car il existe beaucoup plus de documentation sur l'implémentation de systèmes d'exploitation sur Linux que sur Mac. Bien que Mac OS soit un système UNIX, les exécutables générés sur cet environnement n'ont pas le même format que ceux générés sur Linux qui sont au format ELF. Ceci rend le développement d'OS légèrement différent sur Mac OS.

3.2 Technologies

Bien que le système d'exploitation développé devait être sur Rust, certaines parties ont du être faites en assembleur car étant trop bas niveau pour le Rust. Ces éléments seront décrits plus loin dans ce document. `Nasm` a été utilisé pour compiler le code assembleur x86 en elf 32-bit. `Nasm` produit des fichiers objets pouvant être liés à d'autres fichiers objets afin de créer un exécutable.

Rust sera décrit plus en détail dans un prochain chapitre. Ce qu'il faut savoir est que Rust est distribués sous trois versions différentes. La version *stable*, la version *beta* et la version *nightly*. La version *nightly* possède plus de fonctionnalités mais sa stabilité n'est pas garantie. Cette version a été utilisée pendant le développement du projet et l'utilitaire Rustup a été utilisé pour son installation. Cet utilitaire permet de simplifier l'installation de Rust quand on souhaite une version différente de la dernière version stable de Rust.

Lors du développement d'un système d'exploitation type *bare metal*, on souhaite s'affranchir de toute dépendance à une librairie externe. Tout doit être refait depuis le début. Le code est donc compilé sans la bibliothèque standard (std). Rust a tout de même besoin d'une base pour être compilé. Cette base est fournie par la librairie `core`. Cette librairie est minimale et permet de ne définir que les primitives de Rust. Pour gérer les dépendences d'un projet Rust, il est conseillé d'utiliser le gestionnaire de paquets `cargo`. Le problème est que `cargo` ne permet pas de lier la librairie `core` à un projet. Heureusement, un autre utilitaire basé sur `cargo` existe et permet d'installer par défaut la librairie `core` pour des projets sans bibliothèque standard. Cet utilitaire se nomme `xargo` et est utilisé pour compiler le code Rust en fichiers objets

Le compilateur `gcc` a été utilisé pour *linker* les fichiers objet générés par `nasm` et `xargo`. `Gcc` génère un fichier au format ELF. Pour utiliser ce fichier comme un système d'exploitation *bootable*, il faut en faire une image ISO *bootable*. Pour se faire, l'utilitaire `genisoimage` est utilisé, couplé au *bootloader* GRUB.

3.3 Architecture

4 Rust

5 Système d'exploitation de type *bare metal*

5.1 Compilation

5.2 *Linking*

5.3 Processus de *boot*

5.4 Adressage mémoire

5.5 Ports

5.6 Interruptions

5.7 Periphériques

5.7.1 VGA

5.7.2 *Timer*

5.7.3 Clavier

6 Système de fichiers

6.1 Introduction

6.2 Structure

7 Résultats

8 Discussions

8.1 Problèmes rencontrés

8.2 Améliorations possibles

9 Conclusion

10 Références

[1] Rust book first edition. https://doc.rust-lang.org/book/first-edition.

[2] Rust book second edition. https://doc.rust-lang.org/book/second-edition.

[3] Cargo book. https://doc.rust-lang.org/cargo.

[4] Target option. https://doc.rust-lang.org/1.1.0/rustc_back/target/struct.Target.html.

[5] Target i386 example. https://github.com/rust-lang/rust/issues/33497.

[6] ___floatundisf issue. https://users.rust-lang.org/t/kernel-modules-made-from-rust/9191.

[7] Writing an os in rust. https://os.phil-opp.com.

[8] Writing an os in rust (second edition). https://os.phil-opp.com/second-edition.