Boot et premier programme en mode kernel

Préambule

Si ce n'est pas encore fait, vous devez lire la page décrivant les objectifs généraux des séances, le principe pédagogique choisi et le fonctionnement des séances.

La première séance est découpé en 5 étapes :

1. hello_boot
   Nous commençons par un petit programme de quelques lignes en assembleur, placé entièrement dans la région mémoire du boot, qui réalise l'affichage du message "Hello World". C'est un tout tout petit programme, mais pour obtenir l'exécutable, vous devrez utiliser tous les outils de la chaîne de cross-compilation MIPS et pour l'exécuter vous devrez exécuter le simulateur du prototype. C'est simple, mais c'est nouveau pour beaucoup d'entre vous.

2. kinit_asm
   Dans le deuxième programme, nous restons en assembleur, mais nous avons deux fichiers source : (1) le fichier contenant le code de boot et (2) le fichier contenant le code du noyau. Ici, le code du noyau c'est juste une fonction `kinit()`. Cette étape permet de voir comment saute d'un fichier à l'autre. Il utilise aussi une version plus complète du fichier `ldscript` du kernel.

3. kinit_c
   Dans ce troisième programme, nous faisons la même chose que pour le deuxième mais `kinit()` est désormais écrit en langage C. Cela change peut de chose, sauf une chose importante `kinit()` est une

fonction et donc il faut absolument une pile d'exécution.

4. nttys

Le prototype de SoC que nous utilisons pour les TP est configurable. Il est possible par exemple de choisir le nombre terminaux texte (TTY). Par défaut, il y en a un mais, nous pouvons en avoir jusqu'à 4. Nous allons modifier le code du noyau pour s'adapter à cette variabilité.

5. driver

1. Premier programme en assembleur dans la seule section de boot

Codes

```
.section .boot,"ax",@progbits       // def. of a new section: .boot (see https://bit.ly/3gzKWob)
                                    // flags "ax":     a -> allocated means section put in memor
                                    //                 x -> section contains instructions
                                    // type @progbits: contains somethings (not just space)
boot:
    la      $26,    kinit           // get address of kinit() function
    j       $26                     // goto kinit()
```

tp2_hcpu.s

Objectif

- Affichage d'un message sur le terminal avec un programme en assembleur.

Compétences acquises

- Savoir produire un exécutable à partir d'un code en assembleur.
- Savoir comment afficher un caractère sur un terminal.

Questions

- Dans quel fichier se trouve la description de l'espace d'adressage du MIPS ?
- Dans quel fichier se trouve le code de boot et pourquoi l'avoir nommé ainsi ?
- A quelle adresse démarre le MIPS ?
- Que produit le compilateur C quand on utilise l'option -c ?
- Que fait l'éditeur de liens ?
- De quels fichiers a besoin l'éditeur de liens pour fonctionner ?
- Dans quelle section se trouve le code de boot pour le compilateur ?
- Dans quelle section se trouve le message hello pour le compilateur ?
- Dans quelle section se trouve le code de boot dans le code exécutable ?
- Dans quelle région de la mémoire le code de boot est placé ?
- Comment connaît-on l'adresse du registre de sortie du contrôleur de terminal TTY ?
- Comment sait-on que le message est fini et que le programme doit s'arrêter ?

- Pourquoi terminer le programme par un `dead: j dead` ?

2. Saut dans le code du noyau en assembleur

Objectif de l'étape

- Affichage d'un message depuis le code du noyau toujours en assembleur

Compétences acquises

- Comment aller à une adresse définie dans un autre fichier
- Création d'une section dans le code objet produit par le compilateur

Questions

- Quel est le nom de la directive assembleur permettant de déclarer une section

3. Saut dans la fonction kinit() du noyau en langage C

Objectif de l'étape

-

Compétences acquises

-

Questions

-

4. Accès aux registres de contrôle des terminaux TTY

Objectif de l'étape

-

Compétences acquises

-

Questions

-

5. Premier petit pilote pour le terminal

Objectif de l'étape

-

Compétences acquises

-

Questions

-