Programmation noyau et écritures de drivers Linux

Objectifs

- Maîtriser les outils kernel de développement et de mise au point
- Programmer les IO, les interruptions, les timers, le DMA
- Adapter les sources des drivers du marché
- Installer et intégrer les drivers dans un kernel linux
- Gérer les interfaces standard synchrones, asynchrones et ioctl
- Développer les structures des drivers caractères, blocs et réseaux
- Comprendre les spécificités de la version 2.6
- Comprendre le hot-plug et les drivers USB

Matériel

- Un PC Linux par binôme
- Support de cours
- CDROM avec documentation et exercices corrigés

Exercise: *Tous les exercices se font en binôme sur ARM 926 en réseau avec la station de développement des stagiaires*

Pré-requis

- Connaissance de la programmation Linux utilisateur et système (niveau cours NX1)

Course Environment

- Theoretical course
  - PDF course material (in English) supplemented by a printed version for face-to-face courses.
  - Online courses are dispensed using the Teams video-conferencing system.
  - The trainer answers trainees' questions during the training and provide technical and pedagogical assistance.
- Practical activities
  - Practical activities represent from 40% to 50% of course duration.
  - Code examples, exercises and solutions
  - For remote trainings:
    - One Online Linux PC per trainee for the practical activities.
    - The trainer has access to trainees' Online PCs for technical and pedagogical assistance.
    - QEMU Emulated board or physical board connected to the online PC (depending on the course).
    - Some Labs may be completed between sessions and are checked by the trainer on the next session.
  - For face-to-face trainings:
    - One PC (Linux ou Windows) for the practical activities with, if appropriate, a target board.
    - One PC for two trainees when there are more than 6 trainees.
  - For onsite trainings:
    - An installation and test manual is provided to allow preinstallation of the needed software.
    - The trainer come with target boards if needed during the practical activities (and bring them back at the end of the course).
- Downloadable preconfigured virtual machine for post-course practical activities
- At the start of each session the trainer will interact with the trainees to ensure the course fits their expectations and correct if needed

Target Audience

- Any embedded systems engineer or technician with the above prerequisites.

Evaluation modalities

- The prerequisites indicated above are assessed before the training by the technical supervision of the traineein his company, or by the trainee himself in the exceptional case of an individual trainee.
- Trainee progress is assessed in two different ways, depending on the course:
  - For courses lending themselves to practical exercises, the results of the exercises are checked by the trainer while, if necessary, helping trainees to carry them out by providing additional details.
  - Quizzes are offered at the end of sections that do not include practical exercises to verifythat the trainees have assimilated the points presented
- At the end of the training, each trainee receives a certificate attesting that they have successfully completed the course.
  - In the event of a problem, discovered during the course, due to a lack of prerequisites by the trainee a different or additional training is offered to them, generally to reinforce their prerequisites,in agreement with their company manager if applicable.

Plan

Premier jour

Le développement de modules Kernel Linux

- Outils de développement
  - outils de compilation
  - makefile
  - chargement et gestion des versions des modules kernel
  - debug kernel
- Intégration des modules
  - regénération du noyau linux
  - intégration des drivers dans les fichiers de configuration du noyau
  - fichiers spéciaux /dev
- Boot des modules
  - algorithme d'allocation mémoire buddy et slab allocator
  - zones mémoire d'allocations des modules
  - allocation dynamique et au boot

Exercise: *Ecriture d'un module simple*

La modélisation des périphériques Linux (2.6)

- Linux Device Model
- Structure objet du noyau Linux
  - kobject
  - kset
  - subsystem
- Le système de fichiers sysfs
- Le hotplug
  - les scripts de gestion du hot-plug
  - les scripts "udev"

Exercise: *Fabrication d'un objet noyau et de son interface sysfs*
Exercise: *Création de fichiers spéciaux (devices) à l'aide d'udev pour une clef USB*

Deuxième et troisième jours

La structure des drivers Linux

- interfaces fichiers (synchrones bloquantes, asynchrones non bloquantes, directes)
- les différents types de drivers Linux
- drivers caractère
  - différences entre les versions 2.4 et 2.6
  - les routines d'initialisation et de terminaison
  - les routines d'entrées-sorties synchrones, asynchrones et de contrôle
- drivers bloc
  - gestion clusturisée des requêtes et des blocs de transferts
  - points de montage
  - routine de stratégie
- drivers réseau
  - configuration et detection automatique des cartes
  - paramètres du noyau
  - transmission et réception des paquets
  - buffers des sockets
  - résolution des adresses
  - lien avec les protocoles
  - nouveautés 2.6

Exercise: *Ecriture du code d'initialisation d'un pilote*
Exercise: *Ecriture des routines de lecture-écriture d'un pilote logiciel*
- version sans synchronisation
- version synchrone
- version asynchrone

Quatrième jour

Accès au matériel

- protection des espaces
- espaces d'E/S déportées et RAM
- mapping de plages mémoires physiques
- spécificités des interfaces PCI et USB

Exercise: *Ecriture d'un pilote d'entrées/sorties parallèles*

Accès Direct Mémoire (DMA)

- allocation de buffer slave et Master
- transferts direct dans les zones mémoires applicatives

Interruptions et exceptions

- architecture des interruptions sur PC
- initialisation de la IDT (Interrupt Dispatch Table)
- routines d'interruptions fast & slow
  - nested interrupts
  - partage des interruptions PCI

Exercise: *Gestion d'interruptions dans le pilote parallèle*

Timers et horloges

- horloges matérielles
  - gestionnaire d'interruption du timer
  - routine Bottom half du timer
- horloge logicielles
  - exécution différée
  - timers kernel

Exercise: *Gestion de timeouts dans le driver parallèle*
Exercise: *Actions périodiques*

Cinquième jour

Les drivers USB

- Drivers USB host
  - énumération du bus
  - intégration avec le hotplug
  - gestion des URB (USB request block)
- Drivers USB device (gadgets)

Exercise: *Manipulation sur un driver réseau sur bus USB*

Renseignements pratiques

Inquiry : 5 days