

*Le support pour les périphériques est assuré sous Linux par des pilotes (drivers) dont le code se déroule dans le noyau du système d'exploitation. Il est donc nécessaire pour le développeur amené à écrire ou à tester des pilotes de périphériques de maîtriser les concepts propres à la programmation noyau.*

*Ce cours propose une approche originale, s'appuyant sur l'écriture progressive de pilotes de différents types, pour appréhender les mécanismes parfois complexes (préemptibilité, multiprocesseur, support d'architectures différentes, etc.) inhérents au code exécuté en mode noyau.*

*Outre les périphériques classiques (caractère, bloc, réseau), on étudie certains sous-systèmes du noyau tels que les systèmes de fichiers ou l'ensemble USB.*

### Organisation

---

**Durée** : 4 jours.

**Pré-requis** : Connaissance de Linux (utilisateur) et bonne familiarité avec le langage C.

**Conseil cursus** : La formation Linux Temps-Réel et Embarqué représente un bon complément à ce cours.

### Thèmes abordés

---

- **Programmer pour le noyau Linux** : noyau Linux et modules, outils de développement, API et environnement de fonctionnement du noyau ;
- **Périphériques caractère** : principes d'écriture d'un pilote de périphérique, appels-systèmes principaux, accès au matériel, traitement des interruptions, *tasklet*, *workqueue* ;
- **Driver avancé** : attentes, asynchronisme, mémoire, classe de périphériques ;
- **Périphériques en mode bloc** : principes, traitement des requêtes, partitionnement, ordonnancement des entrées-sorties ;
- **Systèmes de fichiers** : principe du *VFS*, inscription d'un nouveau système de fichiers ;
- **Périphériques et protocoles réseau** : interface réseau, entregistrement d'une nouvelle interface, étude de la pile IP ;
- **Périphériques USB** : sous-système USB, enregistrement d'un driver *Interrupt*, études de drivers *Bulk* et *Control*.

► *Plan détaillé au dos...*

### Première journée - Programmer pour le noyau Linux

#### 1ère séquence 9h00-10h30

Noyau Linux et modules : Historique du noyau Linux, licence GPL, développement du noyau. Appels-système, modules.

##### Travaux pratiques

Observation des appels-système invoqués par des applications et commandes utilisateur. Manipulation des modules précompilés.

#### 2ème séquence 10h45-12h15

Outils de développement noyau : Organisation des sources, compilation du noyau et des modules. Programmation de modules du noyau, compilation de modules indépendants. Messages du noyau, dépendances entre modules.

##### Travaux pratiques

Compilation et installation d'un noyau 2.6. Application de patches pour débogage noyau. Écriture de modules simples du noyau. Intégration dans la chaîne de compilation du noyau. Passage de paramètres au boot.

#### 3ème séquence 13h30-15h15

Interface de programmation du noyau : Chaînes de caractères, blocs mémoire, fonctions numériques et conversions. Pilotes de périphériques. Éléments temporels et actions différées. Préemptibilité du noyau 2.6.

##### Travaux pratiques

Écriture d'un module d'horodatage. Chronométrer les phases de boot.

#### 4ème séquence 15h30-17h30

Environnement du noyau : Tâches et processus current. Espaces d'adressage. Dialogue avec /proc.

##### Travaux pratiques

Écriture d'un module d'information sur les structures internes des processus.  
Écriture d'un module d'horodatage via /proc.

### Deuxième journée - Écriture d'un driver

#### 1ère séquence 9h00-10h30

Écriture d'un pilote de périphérique : Principe des pilotes de périphérique. Réservation de numéros majeurs et mineurs. Enregistrement du pilote de périphérique. Fonctions de lecture et écriture. Fonctions de paramétrage. Synchronisation des appels-système.

##### Travaux pratiques

Manipulation des fichiers spéciaux. Réservation de numéro majeur. Enregistrement de périphérique. Écriture d'un driver simple. Implémentation des routines de lecture et écriture.

#### 2ème séquence 10h45-12h15

Accès au matériel et interruptions : Accès simple au matériel. Contextes d'exécution : appel-système et interruption. Gestion des interruptions. Différer un traitement en interruption : tasklet & workqueue. Protection des variables globales.

##### Travaux pratiques

Écriture d'un gestionnaire d'interruption.

#### 3ème séquence 13h30-15h15

Fonctions avancées d'un pilote de périphérique : Attentes d'événements. Multiplexage d'entrées-sorties. Gestion de la mémoire.

##### Travaux pratiques

Création d'un périphérique "file de messages" virtuel.

#### 4ème séquence 15h30-17h30

Modèle de périphérique du noyau 2.6 : Création d'une classe de périphérique.  
DMA : Transferts de données par DMA.

.../...

## Troisième journée - Périphériques bloc et systèmes de fichiers

### 1ère séquence 9h00-10h30

Périphériques en mode bloc : Principe des périphériques bloc. Ecriture d'un driver. Enregistrement du pilote. Déclaration d'un disque générique. Initialisation de la file de requêtes. Requetes sur un driver bloc.

### 2ème séquence 10h45-12h15

Driver bloc avancé : Traitement différé. Partitionnement du disque. Sous-système Block du noyau. Ordonnanceur des entrées-sorties

#### Travaux pratiques

Ecriture progressive d'un pilote de disque virtuel, en s'appuyant sur les exemples fournis dans le cours

### 3ème séquence 13h30-15h15

Virtual File System : Organisation du VFS Structures File, Dentry, Inode, et Super-bloc.

### 4ème séquence 15h30-17h30

Nouveau système de fichiers : Enregistrement. Initialisation du super-bloc. Implémentation des appels-système de lecture et écriture. Utilisation du cache en lecture et en écriture. Communication avec le sous-système Block.

#### Travaux pratiques

Ecriture d'un système de fichiers virtuel, simple en analysant les étapes de transfert des données.

## Quatrième journée - Autres types de périphériques

### 1ère séquence 9h00-10h30

Périphériques réseau : Dépendance des interfaces bas-niveau et des protocoles réseau. Périphérique net\_device Enregistrement d'une interface, activation, émission et réception de paquets. Statistiques d'utilisation d'interface

#### Travaux pratiques

Ecriture progressive d'un driver pour périphérique virtuel permettant l'utilisation du protocole IPv4.

### 2ème séquence 10h45-12h15

Protocoles réseau : Principes de la pile IP. Emission et réception de données. Communications avec l'interface bas-niveau.

#### Travaux pratiques

Examen du trajet des données au sein de la pile IPv4 lors de réception et d'émission de données avec le protocole TCP/IP.

### 3ème séquence 13h30-15h15

Périphériques USB : Organisation du sous-système USB de Linux. Implémentation d'un driver de classe Interrupt : Enregistrement d'un driver. Endpoints et types de dialogues. Communication avec les URB. Traitements des écritures successives rapides. Déconnexions intempestives et accès concurrents. Gestion simultanée de plusieurs périphériques.

#### Travaux pratiques

Ecriture d'un driver pour carte d'entrée-sortie Velleman K8055.

### 4ème séquence 15h30-16h30

Autres classes USB : Exemples de drivers pour les modes Bulk, Control et Isochrones.

### Conclusion

Discussions libres sur l'ensemble des thèmes abordés.

#### Travaux pratiques

Expérimentations libres suivant les demandes des participants.