Linux industriel, temps réel et embarqué

Cours Pratique de 4 jours - 28h
Réf : LXT - Prix 2024 : 2 890CHF HT

Vous découvrirez dans cette formation l'architecture générale d'un système embarqué et mettrez en œuvre les outils de développement industriel disponibles librement sous Linux. Vous implémenterez les mécanismes d'ordonnancement temps partagé et réel souple, vous accéderez au développement temps réel strict avec l'extension LinuxRT ou Xenomai. Vous verrez enfin comment personnaliser le boot d'un système Linux.

OBJECTIFS PÉDAGOGIQUES

À l'issue de la formation l'apprenant sera en mesure de :

Découvrir les outils de développement industriel sous Linux

Maîtriser les mécanismes d'ordonnancement temps partagé et réel souple

Accéder au développement temps réel strict avec l'extension LinuxRT ou Xenomai

Personnaliser le boot d'un système Linux

TRAVAUX PRATIQUES

Les nombreux exercices et études de cas progressifs sont réalisés sur un réseau de serveurs Linux. Tous les programmes réalisés en TP existent sous forme de squelettes que les participants complètent eux-mêmes.

LE PROGRAMME

dernière mise à jour : 08/2018

1) Architecture

- Système informatique ordinaire et système embarqué.
- Contraintes d'un système embarqué.
- Architecture générale d'un système embarqué.
- Démarrage du système, étape de boot.
- Architecture du noyau Linux. Emplacement des sources.
- Démarrage du système, phases de boot (code dépendant, commun).

Travaux pratiques : Détection d'erreur à la compilation, à l'édition des liens, utilisation d'Eclipse/CDT, utilisation d'une chaîne de compilation croisée. Débogage. Détection des fuites mémoire et des débordements de buffers. Test de couverture sur l'exécution d'une application.

2) Développement industriel sous Linux

- Environnement Linux.
- Mode de fonctionnement : utilisateur, superviseur.
- Licences et implications pour le développement industriel.
- Outils de développement libres (compilateur, debugger, outils d'analyse, de trace et de tests).
- Les différents IDE (Integrated Development Environment) : Eclipse...
- Méthodes de compilation avancées.
- La chaîne de compilation croisée.
- La gestion de mémoire.
- La détection des fuites mémoire.
- Le débordement de buffers.

PARTICIPANTS

Développeurs Linux/Unix.

PRÉREQUIS

Bonnes connaissances d'un système Linux/Unix et de la programmation en C.

COMPÉTENCES DU FORMATEUR

Les experts qui animent la formation sont des spécialistes des matières abordées. Ils ont été validés par nos équipes pédagogiques tant sur le plan des connaissances métiers que sur celui de la pédagogie, et ce pour chaque cours qu'ils enseignent. Ils ont au minimum cinq à dix années d'expérience dans leur domaine et occupent ou ont occupé des postes à responsabilité en entreprise.

MODALITÉS D'ÉVALUATION

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques…
Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

MOYENS PÉDAGOGIQUES ET TECHNIQUES

• Les moyens pédagogiques et les méthodes d'enseignement utilisés sont principalement : aides audiovisuelles, documentation et support de cours, exercices pratiques d'application et corrigés des exercices pour les stages pratiques, études de cas ou présentation de cas réels pour les séminaires de formation.
• À l'issue de chaque stage ou séminaire, ORSYS fournit aux participants un questionnaire d'évaluation du cours qui est ensuite analysé par nos équipes pédagogiques.
• Une feuille d'émargement par demi-journée de présence est fournie en fin de formation ainsi qu'une attestation de fin de formation si le stagiaire a bien assisté à la totalité de la session.

MODALITÉS ET DÉLAIS D'ACCÈS

L'inscription doit être finalisée 24 heures avant le début de la formation.

ACCESSIBILITÉ AUX PERSONNES HANDICAPÉES

Vous avez un besoin spécifique d'accessibilité ? Contactez Mme FOSSE, référente handicap, à l'adresse suivante psh-accueil@orsys.fr pour étudier au mieux votre demande et sa faisabilité.

3) Ordonnancement temps partagé et réel souple

- Précision des mesures horaires et des attentes.
- Ordonnancement temps partagé. Les règles de la préemptibilité.
- Le fonctionnement général de l'ordonnanceur, priorités et partage du CPU.
- Quand le noyau devient-il préemptible ?
- Gestion du temps et précision des timers.
- Temps réel souple Posix.1b : principes.
- Configuration de l'ordonnancement des processus et des threads.
- Problèmes algorithmiques liés au temps réel.
- L'ordonnanceur Linux : noyaux 3.x

Travaux pratiques : Création et gestion de processus. Priorités et partage du CPU. Examen du fonctionnement de l'ordonnanceur. Basculement de processus en temps réel. Vérification de la précision des timers et des sommeils.

4) Temps réel strict - Extension Xenomai

- Principe du temps réel strict.
- Vue d'ensemble de l'extension Xenomai.
- Concepts de temps réel strict : principe des micro-noyaux Adeos, Xenomai, LinuxRT.
- Installation et API de Xenomai.
- Utilisation de l'extension LinuxRT.
- Ordonnancement temps réel strict en mode utilisateur.
- Interruptions (activation, désactivation...).
- Protection contre les interruptions.
- Gestion des communications.
- Présentation de l'API de Xenomai, installation de Xenomai.
- La gestion des tâches temps réel strict.

Travaux pratiques : Installation de Xenomai. Création de tâches temps réel strict. Gestion des communications. Ecriture d'un gestionnaire d'interruption, de processus ordonnancé en temps réel strict. Installation et utilisation de Xenomai.

5) Environnements restreints, systèmes embarqués

- Problématique des systèmes embarqués.
- Système LinuxRT, Xenomai : API, développement.
- Linux embarqué : choix d'une version du noyau.
- Bibliothèques système (Newlib, DietLibc).
- Applications et utilitaires à embarquer.
- Interface utilisateur.
- Interfaces graphiques optimisées (directfb, etc).
- Présentation et configuration d'un chargeur de démarrage.
- Générer un noyau réduit. Généralité sur le système de fichiers.
- Installation de la chaîne de compilation.

Travaux pratiques : Installation, compilation d'une application personnalisée LinuxRT, Xenomai et d'un noyau de taille réduite. Création d'un système de fichiers. Incorporation d'applications minimales. Ecriture d'application utilisant une interface par Leds ou afficheur LCD. Installation d'un serveur HTTP embarqué.

6) Personnalisation du boot du système

- Les différentes phases de boot (mise sous tension, Bios, chargeur (Grub, UBoot...) du noyau.
- Le rôle du processus Init. Le niveau d'exécution.
- Le contenu du processus Init.
- Le remplacement du processus Init par une version personnalisée.
- Initialisation depuis l'espace utilisateur.

Travaux pratiques : Création et personnalisation d'un disque initrd. Remplacement du processus Init par une version personnalisée et remplacement par un script Shell.

LES DATES

CLASSE À DISTANCE
2024 : 16 juil., 08 oct.