logilin
Logilin formations

Linux embarqué avec Yocto Project

L'utilisation croissante de Linux dans les systèmes embarqués va de pair avec un besoin de fiabilité et de pérennité de l'installation. Pour cela ***Yocto Project*** *est aujourd'hui le système de construction le plus adapté aux nécessités de l'environnement industriel.*

Ce cours vous propose de ***découvrir les rouages de Yocto Project****, d'en maîtriser l'utilisation courante (génération et personnalisation d'images Linux embarqué) afin de développer des applications pour Linux embarqué.*

Si vous avez déjà une bonne pratique de Yocto Project, nous vous conseillerons plutôt notre cours « Yocto Project Avancé », qui explore en profondeur des points supplémentaires de l'utilisation de Yocto (classes, distro, devtool, machine, device tree, wks, etc.)

Organisation

Audience

Nous limitons habituellement le nombre de stagiaires dans nos sessions à 4 personnes au maximum pour garantir des échanges fluides et conviviaux.

Les sessions à distance se déroulent sur **plateforme Zoom**. Le seul matériel nécessaire est un ordinateur avec une connexion Internet et un micro. Nous conseillons un ensemble casque + micro pour limiter le bruit de fond. Nous suggérons également l'emploi d'une webcam si l'environnement le permet.

Pré-requis

Une connaissance de Linux (niveau utilisateur) est nécessaire et une certaine aisance avec la ligne de commande du shell est recommandée.

Durée

3 jours (21 heures)

Travaux pratiques

Les exercices se déroulent sur des PC Linux Ubuntu accessibles à distance (connexion SSH / PuTTY / Tera Term) et émulateur Qemu. Les démonstrations sont présentées sur cartes Raspberry Pi 4.

Thèmes abordés

Créer un système Linux embarqué avec Yocto Project : environnement Linux embarqué, production d'une image, composition d'un système embarqué,

Production d'un BSP Personnalisé : Analyse du système produit, personnalisation et configuration du contenu.

Configuration avancée : extension de recettes, utilisation de *devtool*, *kernel* et *device tree*.

Intégration du code métier : intégration de scripts, développement applicatif embarqué, intégration dans l'image

Plan détaillé

I – Créer un système Linux embarqué avec Yocto Project

Environnement Linux embarqué

Concepts, composant, outils de génération Buildroot et Yocto Project.

Production d'une image standard

Environnement de travail, Poky, configuration, outil Bitbake, *layers* spécifiques.

Composition d'un système Linux embarqué

Matériel, *bootloader*, noyau Linux, processus init, scripts de démarrage.

Travaux pratiques : préparation de l'environnement, production d'une image pour émulateur Qemun Arm, pour pour Raspberry Pi, démarrage et test des images produites.

II – Production d'un BSP personnalisé

Analyse du système produit

Connexion, systèmes de fichiers, arborescence standard, *boot*.

Personnalisation du système

Recette d'image personnelle, administration , syntaxe des recettes, manipulation des variables.

Configuration du contenu

Ajout de packages création d'une *distro*, configuration de Busybox, principe des *features*.

Travaux pratiques : création d'un *layer* personnalisé, d'une image, d'une *distro*, ajout de packages de Poky, d'Open Embedded, configuration de Busybox, utilisation d'une fonctionnalité d'image.

III - Configuration avancée du système

Extension de recettes

Fichiers *.bbappend*, surcharge de fichiers, configuration réseau statique, *patch* sur un fichier de recette.

Utilisation de *devtool*

Recherche d'information, création de *patch*, respect des licences libres.

Noyau Linux et *Device Tree*

Choix, type et version du *kernel*, paramétrage, configuration du *device tree*.

Travaux pratiques : remplacement d'un fichier, création d'un *patch* sur un fichier source, configuration du noyau, *patch* pour le *kernel*, *patch* pour le *device tree*.

IV - Intégration du code métier

Intégration de scripts personnalisés

Scripts shell, scripts Python, écriture de recette.

Développement applicatif embarqué

Principes, *cross-compilation*, SDK, débogage distant.

Intégration dans l'image

Écriture de recette, lancement de l'application au démarrage du système.

Travaux pratiques : écriture et intégration de scripts shell et Python, extraction et utilisation du SDK, débogage et validation du code métier, écriture de recettes pour applicatif C/C++, lancement au boot.

Conclusion

Discussions libres sur l'ensemble des thèmes abordés.

Travaux pratiques : expérimentations libres suivant les demandes des stagiaires.