

Formation développement Linux embarqué avec Yocto Project et OpenEmbedded

Séminaire en ligne, 4 sessions de 4 heures

Dernière mise à jour : 20 May 2024

Titre	Formation développement Linux embarqué avec Yocto Project et OpenEmbedded
Objectifs opérationnels	<ul style="list-style-type: none">• Être capable de comprendre le rôle et le principe d'un build system Linux embarqué, et comparer Yocto Project/OpenEmbedded aux autres outils offrant des fonctionnalités similaires.• Être capable de configurer et de réaliser la compilation d'un système Linux embarqué simple avec Yocto, et d'installer le résultat sur une plateforme embarquée.• Être capable d'écrire ou d'étendre des recettes de paquets, pour vos propres paquets ou personnalisations.• Être capable d'utiliser des <i>layers</i> de recettes existants, et de créer votre propre nouveau <i>layer</i>.• Être capable d'intégrer le support pour votre carte embarqué dans un <i>layer BSP</i>.• Être capable de créer des images personnalisées.• Être capable d'utiliser le SDK du Yocto Project pour développer des applications.• Être capable d'utiliser devtool pour développer une recette.
Durée	Quatre demi-journées - 16 h (4 h par demi-journée)
Méthodes pédagogiques	<ul style="list-style-type: none">• Présentations animées par le formateur, par visioconférence. Les participants peuvent poser des questions à tout instant.• Démonstrations pratiques réalisées par le formateur, basés sur les travaux pratiques de la formation, par vidéo-conférence. Les participants peuvent poser des questions à tout instant. Optionnellement, les participants qui ont accès aux accessoires matériels de la formation peuvent reproduire par eux-même les travaux pratiques.• Messagerie instantanée pour questions entre les sessions (réponse sous 24h, hors week-end et jours fériés)• Version électronique des supports de présentation, des instructions et des données de travaux pratiques. Les supports sont librement disponibles sur https://bootlin.com/doc/training/yocto.



Formateur	Un des ingénieurs mentionnés sur : https://bootlin.com/training/trainers/
Langue	Présentations : Français Supports : Anglais
Public visé	Sociétés et ingénieurs intéressés par l'utilisation de Yocto Project pour construire leur système Linux embarqué.
Pré-requis	<ul style="list-style-type: none">• Connaissance et pratique des commandes UNIX ou GNU/Linux : les participants doivent être à l'aise avec l'utilisation de la ligne de commande Linux. Les participants manquant d'expérience sur ce sujet doivent se former par eux-mêmes, par exemple en utilisant nos supports de formation disponible à l'adresse bootlin.com/blog/command-line/.• Expérience minimale en développement Linux embarqué : les participants doivent avoir une compréhension minimale de l'architecture d'un système Linux embarqué : rôle du noyau Linux par rapport à l'espace utilisateur, développement d'applications espace utilisateur en C. Suivre la formation <i>Linux embarqué</i> de Bootlin, disponible sur bootlin.com/training/embedded-linux/, permet de remplir ce pré-requis.• Niveau minimal requis en anglais : B1, d'après le <i>Common European Framework of References for Languages</i>, pour nos sessions animées en anglais. Voir bootlin.com/pub/training/cefr-grid.pdf pour une auto-évaluation.
Équipement nécessaire	<ul style="list-style-type: none">• Ordinateur avec le système d'exploitation de votre choix, équipé du navigateur Google Chrome ou Chromium pour la conférence vidéo.• Une webcam et un micro (de préférence un casque avec micro)• Une connexion à Internet à haut débit
Modalités d'évaluation	Seuls les participants qui auront assisté à l'intégralité des journées de formation, et qui auront obtenu plus de 50% de réponses correctes à l'évaluation finale recevront une attestation individuelle de formation de la part de Bootlin.



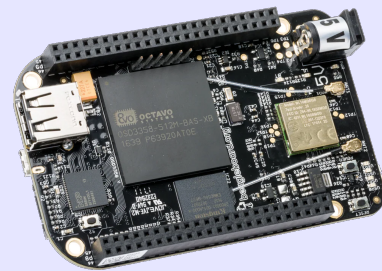
Handicap

Les participants en situation de handicap qui ont des besoins spécifiques sont invités à nous contacter à l'adresse training@bootlin.com afin de discuter des adaptations nécessaires à la formation.

Matériel, première option

Carte BeagleBone Black

- Un processeur ARM AM335x de Texas Instruments (à base de Cortex-A8), avec accélération 3D, etc.
- 512 Mo de RAM
- 2 Go de stockage eMMC embarqué sur la carte (4 Go avec la révision C)
- USB hôte et device
- Sortie HDMI
- Connecteurs à 2 x 46 broches, pour accéder aux UARTs, aux bus SPI, aux bus I2C, et à d'autres entrées/sorties du processeur.



Matériel, deuxième option

Une de ces cartes de STMicroelectronics : **STM32MP157A-DK1**, **STM32MP157D-DK1**, **STM32MP157C-DK2** ou **STM32MP157F-DK2**

- Processeur STM32MP157, double Cortex-A7, de STMicroelectronics
- Alimentée par USB
- 512 Mo DDR3L RAM
- Port Gigabit Ethernet port
- 4 ports hôte USB 2.0
- 1 port USB-C OTG
- 1 connecteur Micro SD
- Debugger ST-LINK/V2-1 sur la carte
- Connecteurs compatibles Arduino Uno v3
- Codec audio
- Divers : boutons, LEDs
- Écran LCD tactile (uniquement sur cartes DK2)





1^{ère} demi-journée

Cours - Introduction aux outils de compilation de systèmes Linux embarqué

- Vue d'ensemble de l'architecture d'un système Linux embarqué
- Méthodes pour compiler un système de fichiers
- Utilité des outils de compilation

Cours - Vue d'ensemble de Yocto Project et du système de référence Poky

- Présentation de l'outil de compilation Yocto / OpenEmbedded et de sa terminologie.
- Vue d'ensemble du système de référence Poky

Cours - Bases de l'utilisation de Yocto Project

- Mise en place du répertoire de travail et de l'environnement
- Configuration de l'outil de compilation
- Compilation de l'image d'un système de fichiers racine

Démo 1 - 1^{ère} compilation avec Yocto Project

- Téléchargement du système de référence Poky
- Configuration de l'outil de compilation
- Compilation d'une image système

Cours - Utilisation de Yocto Project - Notions de base

- Structure des fichiers générés

Démo 1 - Flasher et booter

- Flasher et booter l'image du système sur la carte



2^{ème} demi-journée

Cours - Utilisation de Yocto Project - Utilisation avancée

- Assignment des variables, opérateurs et surcharge
- Paquetages : variantes de paquetages
- Options de la commande bitbake

Démo 2 - Utilisation de NFS et configuration de la compilation

- Configurer la carte pour démarrer via NFS
- Rajouter un paquetage au système de fichiers racine
- Apprendre à utiliser le mécanisme PREFERRED_PROVIDER
- Se familiariser avec les options de la commande bitbake

Cours - Écriture de recettes - Fonctionnalités de base

- Recettes : vue d'ensemble
- Organisation d'un fichier de recette
- Application de patches
- Exemples de recettes

Démo 3 - Ajouter la compilation d'une application

- Création d'une recette pour *nInvaders*
- Résolution de problèmes liés à la recette
- Résolution de problèmes liés à la compilation croisée
- Ajout d'*nInvaders* à l'image finale

Cours - Écriture de recettes - Fonctionnalités avancées

- Extension et redéfinition de recettes
- Paquetages virtuels
- Familiarisation avec les classes
- Inclusion d'exemples avec BitBake
- Mise au point des recettes
- Configuration de l'utilisation du réseau par BitBake



3^{ème} demi-journée

Cours - Layers

- Ce que sont les *layers*
- Où trouver les *layers*
- Création d'un *layer*

Démo 4 - Écriture d'un layer

- Apprendre à écrire un *layer*
- Ajouter le *layer* à la compilation
- Inclure *nInvaders* dans le nouveau *layer*

TP 5 - Étendre une recette

- Étendre la recette pour le noyau pour rajouter des patches
- Configurer le noyau pour compiler le pilote du nunchuk
- Modifier la recette *nInvaders* pour rajouter des patches
- Jouer avec *nInvaders*

Cours - Écriture d'un BSP

- Introduction aux layers BSP
- Ajout d'une nouvelle machine
- Configuration du chargeur de démarrage
- Linux : la classe `kernel.bbclass` et la recette `linux-yocto`

Démo 6 - Création d'une configuration spécifique pour une machine

- Créer une nouvelle configuration de machine
- Compiler une image pour la machine

Cours - Layers de distribution

- Configuration d'une distribution
- Layers de distribution



Cours - Images

- Écriture d'une recette d'image
- Types d'images
- Écriture et utilisation de groupes de paquets

Démo 7 - Création d'une image sur mesure

- Rajouter une recette de base pour une image
- Sélectionner les fonctionnalités et les paquets de l'image
- Ajouter un groupe de paquets sur mesure
- Ajouter une variante d'image pour le débogage

4^{ème} demi-journée

Cours - Écriture de recettes - Pour aller plus loin

- Le sysroot de chaque recette
- Utilisation de code Python code dans les méta-données
- Drapeaux de variables
- Fonctionnalités de paquets et PACKAGEDCONFIG
- Fonctionnalités conditionnelles
- Découpage de paquets
- Détails sur les dépendances

Cours - Licences

- Gestion de licences open source

Cours - SDK pour le projet Yocto

- Objectifs du SDK
- Compilation et personnalisation d'un SDK
- Utilisation d'un SDK pour le projet Yocto

Démo 8 - Développement d'une application au moyen du SDK de Poky

- Compilation d'un SDK
- Utilisation du SDK



Cours - Devtool

- Présentation devtool
- Cas d'utilisation de devtool

Démo 9 - Utilisation de devtool

- Création d'une nouvelle recette
- Modification d'une recette pour ajouter un nouveau patch
- Mettre à jour une recette pour prendre en charge une nouvelle version

Cours - Gestion automatique de layers

- Gestion automatique de layers

Cours - Gestion de paquetages à l'exécution

- Introduction à la gestion de paquetages à l'exécution
- Configuration de la compilation
- Configuration d'un serveur de paquetages
- Configuration de la machine cible

Questions / réponses

- Questions et réponses avec les participants à propos des sujets abordés.
- Présentations supplémentaires s'il reste du temps, en fonction des demandes de la majorité des participants.

Temps supplémentaire possible

Du temps supplémentaire (jusqu'à 4 heures) pourrait être proposé si le programme ne tenait pas en 4 demi-journées, selon le temps passé à répondre aux questions des participants.