

Formation développement Linux embarqué avec Yocto Project et OpenEmbedded

Séminaire de formation en ligne

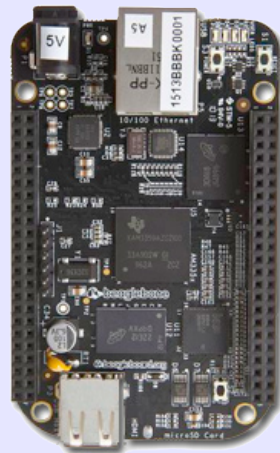
Titre	Formation développement Linux embarqué avec Yocto Project et OpenEmbedded
Aperçu	Comprendre l'architecture de Yocto Project Utilisation pour compiler un système de fichiers et exécuter celui-ci sur votre plateforme matérielle Étendre des recettes (<i>recipes</i>) existantes et en écrire de nouvelles Création de <i>layers</i> Intégration de votre matériel dans un <i>BSP</i> Création d'images sur mesure Développement applicatif à l'aide du SDK de Yocto Project
Supports	Vérifiez que le contenu de la formation correspond à vos besoins : https://bootlin.com/doc/training/yocto .
Durée	Cinq demi-journées - 20 h (4 h par demi-journée) 80% de présentations et 20% de démonstrations.
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	Connaissance de Linux embarqué , sujet couvert par notre formation Linux embarqué (https://bootlin.com/training/embedded-linux/) Connaissance et pratique des commandes UNIX ou GNU/Linux Les personnes n'ayant pas ces connaissances peuvent s'autoformer, par exemple en utilisant nos supports de formation disponibles en ligne : (https://bootlin.com/blog/command-line/)
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
Supports	Version électronique des présentations, des instructions et des données pour les démos.



Matériel

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.



1^{ère} demi-journée

Cours - Introduction aux outils de compilation de systèmes Linux embarqué

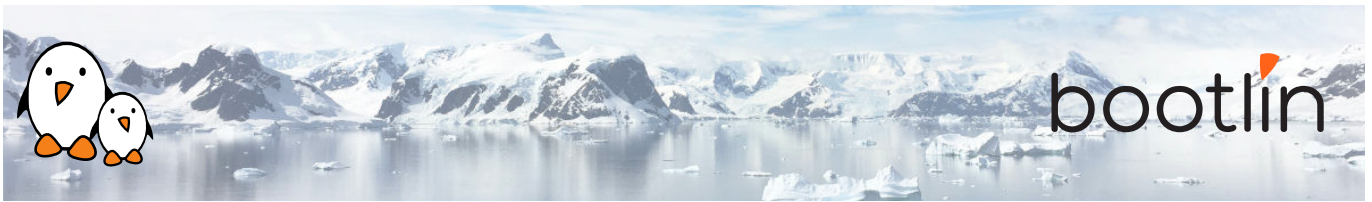
- Vue d'ensemble de l'architecture d'un système Linux embarqué
- Méthodes pour compiler un système de fichiers
- Utilité des outils de compilation

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

- Organisation des sources du projet
- Création d'un système de fichiers avec Yocto Project

Démo - 1^{ère} compilation avec Yocto Project

- Téléchargement du système de référence Poky
- Compilation d'une image système



Cours - Utilisation de Yocto Project - Notions de base

- Structure des fichiers générés
- Flasher et installer l'image du système

2^{ème} demi-journée

Démo - Flasher et booter

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

Cours - Utilisation de Yocto Project - Utilisation avancée

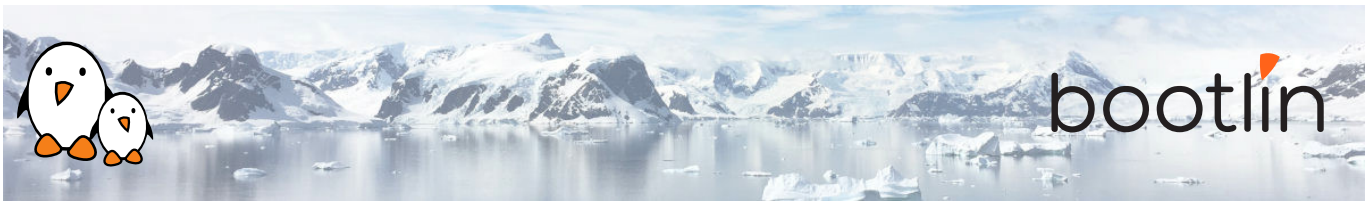
- Configuration de la compilation
- Personnalisation de la sélection de paquets

Démo - Utilisation de NFS et configuration de la compilation

- Configurer la carte pour démarrer via NFS
- Apprendre à utiliser le mécanisme `PREFERRED_PROVIDER`

Cours - Écriture de recettes - Fonctionnalités de base

- Écriture d'une recette minimale
- Ajout de dépendances
- Organisation du développement avec *bitbake*



3^{ème} demi-journée

Démo - Ajouter la compilation d'une application

- Création d'une recette pour *nInvaders*
- Ajout d'*nInvaders* à l'image finale

Cours - Écriture de recettes - Fonctionnalités avancées

- Extension et redéfinition de recettes
- Rajouter des étapes au processus de compilation
- Familiarisation avec les classes
- Analyse d'exemples
- Logs
- Mise au point des dépendances

Démo - Apprendre à configurer les paquetages

- Extension d'une recette pour ajouter des fichiers de configuration
- Utilisation de `ROOTFS_POSTPROCESS_COMMAND` pour modifier le système de fichier final
- Étude des dépendances entre paquetages

Cours - Layers

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

Démo - Écriture d'un layer

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



4^{ème} demi-journée

Cours - Écriture d'un BSP

- Extension d'un BSP existant
- Ajout d'une nouvelle machine
- Chargeurs de démarrage
- Linux et la recette linux-yocto
- Ajouter un type d'image personnalisé

Démo - Mise en oeuvre de modifications du noyau

- Extension de la recette pour le noyau pour ajouter le pilote pour le Nunchuk
- Configurer le noyau pour compiler le pilote du Nunchuk
- Jouer à *nInvaders*

Cours - Création d'une image sur mesure

- Écriture d'une recette d'image
- Ajouter des utilisateurs et des groupes
- Ajouter une configuration personnalisée
- Écrire et utiliser des groupes de recettes de paquetages

5^{ème} demi-journée

Démo - Création d'une image sur mesure

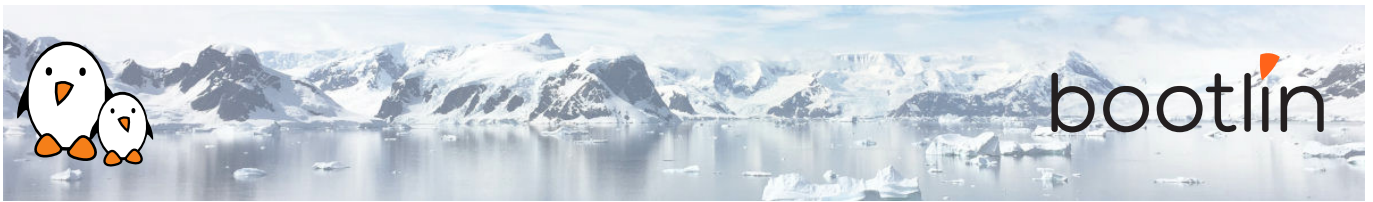
- Écrire une recette d'image personnalisée
- Ajouter *nInvaders* à l'image sur mesure

Cours - Création et utilisation d'un SDK

- Comprendre l'utilité d'un SDK pour le développeur d'applications
- Construire un SDK pour l'image sur mesure

Démo - Expérimentations avec le SDK

- Construction d'un SDK
- Utilisation du SDK de Yocto Project



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.