Formation développemment Linux embarqué avec Buildroot

Formation en présentiel

Durée de la formation

3 jours – 24 h

Langue

Transparents	Anglais
Présentation	Français Anglais

Formateur

Un des ingénieurs suivants

- Thomas Petazzoni

Contact

training@bootlin.com
+33 4 84 25 80 96

Public visé

Sociétés qui utilisent déjà Buildroot ou qui sont intéressées par l'utiliser pour construire leurs systèmes Linux embarqué.

Objectifs opérationnels

- Être capable de comprendre le principe d'un build system Linux embarqué, et comparer Buildroot aux autres outils offrant des fonctionnalités similaires.
- Être capable de créer un système Linux embarqué simple avec Buildroot : créer une configuration, lancer la compilation, installer le résultat sur une plateforme embarquée.
- Être capable d'ajuster la configuration de Buildroot pour construire un système Linux embarqué adapté à des besoins spécifiques : choix de la chaîne de compilation croisée, gestion de la configuration du noyau Linux, personnalisation du système de fichiers racine.
- Être capable de créer de nouveaux paquets dans Buildroot pour intégrer des applications et bibliothèques supplémentaires dans le système Linux embarqué.
- Être capable d'utiliser les outils proposés par Buildroot pour gérer et analyser le build : suivi des vulnérabilités, comformité aux licences open-source, etc.
- Être capable de développer et débugger des applications user-space Linux dans un contexte où Buildroot est utilisé.
- Être capable d'interagir avec la communauté open-source du projet Buildroot et de comprendre le fonctionnement interne de Buildroot.

Prérequis

- **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.
- **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 Linux embarqué de Bootlin permet de remplir ce pré-requis.
- **Niveau minimal requis en anglais : B1**, d'après le *Common European Framework of References for Languages*, pour nos sessions animées en anglais. Voir la grille CEFR pour une auto-évaluation.

Méthodes pédagogiques

- Présentations animées par le formateur : 40% de la durée de formation
- Travaux pratiques réalisés par les participants : 60% de la durée de formation
- Version électroniques de supports de présentation, des instructions et des données de travaux pratiques. Les supports sont librement disponibles ici.

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.

Équipement nécessaire

Pour les sessions en présentiel délivrées chez nos clients, notre client devra fournir :

- Projecteur vidéo
- Un ordinateur sur chaque bureau (pour une ou deux personnes), avec au moins 16 Go de RAM et Ubuntu Linux 24.04 installé dans une partition dédiée d'au moins 30 Go.
- Les distributions autres que Ubuntu Linux 24.04 ne sont pas supportées, et l'utilisation de Linux dans une machine virtuelle n'est également pas supportée.
- Connexion à Internet rapide et sans filtrage : au moins 50 Mbit/s de bande passante en téléchargement, et pas de filtrage des sites Web et protocoles.
- Les ordinateurs contenant des données importantes doivent être sauvegardés avant d'être utilisés dans nos sessions.

Pour les sessions en présentiel organisés dans les locaux de Bootlin, Bootlin fournit l'ensemble de l'équipement.

Plateforme matérielle pour les travaux pratiques

Plateforme STM32MP1

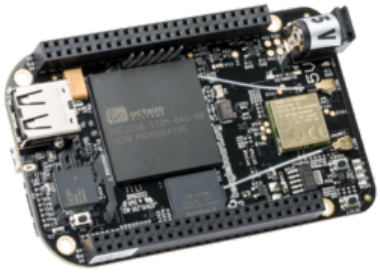
Une de ces cartes de STMicroelectronics : **STM32MP157A-DK1**, **STM32MP157D-DK1**, **STM32MP157C-DK2** ou **STM32MP157F-DK2**

- 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)

BeagleBone Black

Carte **BeagleBone Black** ou **BeagleBone Black Wireless**

- Un processeur ARM AM335x de Texas Instruments (à base de Cortex-A8), avec accélération 3D, etc.
- 512 Mo de RAM
- 2 ou 4 Go de stockage eMMC
- 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.
- Ethernet ou WiFi



Programme de la formation

Jour 1 - Matin

Cours	Introduction à Buildroot et aux systèmes de build	▪ Architecture générale d'une système Linux embarqué ▪ Choix entre systèmes de build et distributions binaires ▪ Rôle d'un système de build ▪ Comparaison des systèmes de build existants
Cours	Présentation de Buildroot	▪ Points clés autour du projet ▪ Téléchargement des sources de Buildroot ▪ Configuration simple de Buildroot ▪ Exécution d'une premières compilation
TP	Utilisation simple de Buildroot	▪ Téléchargement et configuration de Buildroot ▪ Configurer et compiler un système simple avec Buildroot pour un système embarqué ▪ Flasher et tester le système généré par Buildroot
Cours	Gestion de la compilation et de la configuration	▪ Compilation en dehors des sources ▪ Utiliser et créer des fichiers *defconfigs* ▪ Fragments de *defconfigs* ▪ Autres astuces pour la compilation

Jour 1 - Après-midi

Cours	Sources de Buildroot et arborescence des fichiers générés	▪ Détails sur l'organisation du code source de Buildroot ▪ Détails sur l'arborescence des fichiers générés
Cours	Chaînes de compilation *toolchains* dans Buildroot	▪ Les différents possibilités d'usage de chaînes de compilation dans Buildroot. ▪ Tour d'horizon des options liées aux chaînes de compilation. ▪ Utilisation de chaînes des compilation binaires, comme celles de Bootlin. Détails sur les fonctionnalités *multilib* et l'intégration des toolchains dans Buildroot. ▪ Génération de toolchains sur mesure avec *Crosstool-NG*, et leur utilisation comme chaînes externes.
Cours	Gestion de la configuration du noyau Linux	▪ Charger, modifier et sauvegarder la configuration du noyau.
Cours	Construction du système de fichier racine dans Buildroot	▪ Comprendre comment Buildroot construit le système de fichiers racine : *skeleton*, installation de composants, *overlays*, scripts *post-build* et *post-image*. ▪ Personnalisation du contenu du système de fichiers ▪ Configuration du système : sélection de la *console*, plusieurs méthode de gestion de `/dev`, les différentes implémentations d'`init`, etc. ▪ Comprendre comment Buildroot génère les images de systèmes de fichiers.
TP	Personnalisation du système de fichiers	▪ Exploration des fichiers générés ▪ Personnalisation du système de fichiers racine en utilisant un *rootfs overlay* ▪ Personnaliser le noyau avec des correctifs et des options de configuration supplémentaires ▪ Rajout de nouveaux composants ▪ Utilisation de fichiers *defconfig* et compilation en dehors des sources.

Jour 2 - Matin

Cours	Infrastructure de téléchargement dans Buildroot	▪ Méthodologie de téléchargement ▪ Site primaire et sites de backup, compilation en mode déconnecté ▪ Téléchargement via systèmes de contrôle de versions, vérification d'intégrité ▪ Cibles *make* en rapport avec les téléchargements
Cours	Introduction à GNU Make	▪ Éléments de base des règles de make ▪ Définition et utilisation de variables ▪ Conditions et fonctions ▪ Écriture de recettes
Cours	Intégration de nouveaux composants dans Buildroot	▪ Comment rajouter de nouveaux paquetages au système de configuration de Buildroot ▪ Comprendre les différentes infrastructures de paquetages : pour des composants *generic*, *autotools*, *CMake*, *Python* et autres ▪ Écriture un fichier `Config.in` pour un composant : comment exprimer des dépendances vers d'autres composants, vers des options de toolchains, etc. ▪ Détails sur l'écriture d'une recette pour un composant : description de l'emplacement du code source, de la méthode de téléchargement, de configuration, de compilation et d'installation, gestion des dépendances, etc.
TP	Nouveaux composants dans Buildroot	▪ Création d'un nouveau paquetage pour *nInvaders* ▪ Comprendre comment rajouter des dépendances ▪ Ajouter des correctifs pour *nInvaders* pour prendre en charge le contrôle via un *Nunchuk*

Jour 2 - Après-midi

Cours	Notions avancées sur les paquetages	▪ Rapport de licences ▪ Prise en charge des correctifs : ordre d'application et format, répertoire global pour les correctifs, etc. ▪ Utilisateur, droit d'accès, tables de fichiers devices ▪ Script d'init et fichiers unitaires pour systemd ▪ Scripts de configuration ▪ Compréhension des *hooks* ▪ Surcharger des commandes ▪ Gestion des paquetages legacy ▪ Paquetages virtuels
TP	Paquetages avancés	▪ Packager une application avec une dépendance obligatoire et une dépendance optionnelle ▪ Packager une bibliothèque, hébergée sur GitHub ▪ Utilisation de *hooks* pour ajuster les paquetages ▪ Rajouter un correctif à un paquetage

Jour 3 - Matin

Cours	Analyse d'une compilation : licences, dépendances, temps de construction	▪ Utilisation de l'infrastructure de gestion des informations légales ▪ Représentation graphique des dépendances entre paquetages ▪ Collecte d'informations et représentation du temps de compilation
Cours	Sujets avancés	▪ BR2_EXTERNAL pour stocker des personnalisations à l'extérieur des sources de Buildroot ▪ Cibles make spécifiques pour les paquetages ▪ Comprendre les recompilations ▪ Astuces pour compiler plus vite

TP	Sujets avancés	▪ Utilisation des capacités de génération de graphes de temps de compilation ▪ Génération de graphes de dépendances ▪ Utilisation du rapport sur les licences, et ajout d'informations légales à vos propres paquetages ▪ Utilisation de `BR2_EXTERNAL`

Jour 3 - Après-midi

Cours	Développement applicatif avec Buildroot	▪ Utilisation de Buildroot pendant le développement d'applications ▪ Utilisation de l'environnement de Buildroot pour compiler des applications en dehors de Buildroot ▪ Générer un SDK pour d'autres développeurs ▪ Débug à distance avec Buildroot
TP	Développement applicatif avec Buildroot	▪ Compiler et exécuter votre propre application ▪ Débug à distance de votre application ▪ Utilisation de `<pkg>_OVERRIDE_SRCDIR`
Cours	Comprendre les mécanismes internes de Buildroot	▪ Description détaillée du processus de compilation de Buildroot : toolchain, paquetages, construction du système de fichiers race, fichiers *stamp*, etc. ▪ Comprendre les paquetages virtuels.
Cours	Obtenir de l'aide et s'impliquer	▪ Obtenir de l'assistance technique : *Bugzilla*, *liste de discussion*, *IRC* ▪ Contribuer : comprendre le processus de développement, comment soumettre des correctifs