

Formation optimisation du temps de démarrage de Linux embarqué

Formation sur site, 3 jours

Titre	Formation optimisation du temps de démarrage de Linux embarqué
Objectifs opérationnels	<ul style="list-style-type: none">• Être capable des outils et techniques pour mesurer le temps de démarrage d'un système embarqué.• Être capable de réduire le temps de démarrage au niveau de l'initialisation de l'espace utilisateur Linux.• Être capable de réduire le temps de démarrage au niveau de l'initialisation du noyau Linux.• Être capable de réduire le temps de démarrage au niveau de l'initialisation du chargeur de démarrage.• Être capable de mettre en oeuvre d'autres techniques avancées et alternatives d'optimisation du temps de démarrage.
Supports	Vérifiez que le contenu de la formation correspond à vos besoins : https://bootlin.com/doc/training/boot-time .
Durée	Trois jours - 24 h (8 h par jour) 25% de présentations et 75% de travaux pratiques.
Formateur	Un des ingénieurs mentionnés sur : https://bootlin.com/training/trainers/
Langue	Présentations : Français Supports : Anglais
Public visé	Sociétés et ingénieurs développeurs de systèmes Linux embarqués. Personnes offrant de l'assistance à de tels développeurs.
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 doivent s'autoformer, par exemple en utilisant nos supports de formation disponibles en ligne : (https://bootlin.com/blog/command-line/)

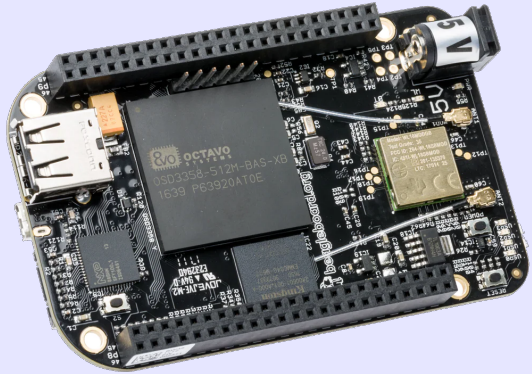


Équipement nécessaire	Pour les sessions sur site uniquement Le matériel est fourni par Bootlin durant les sessions inter-entreprises <ul style="list-style-type: none">• Projecteur vidéo• Un ordinateur sur chaque bureau (pour une ou deux personnes), avec au moins 8 Go de RAM, un processeur au moins équivalent à un Intel Core i5, et Ubuntu Linux installé dans une partition dédiée d'au moins 40 Go. L'utilisation de Linux dans une machine virtuelle n'est pas supportée, en raison de problèmes avec la connexion au matériel.• Nous avons besoin d'Ubuntu Desktop 20.04 (Xubuntu et autres variantes fonctionnent également). Nous ne supportons pas d'autres distributions, car nous ne pouvons tester toutes les versions des paquets.• Connexion à Internet (directe ou par le proxy de l'entreprise).• Les ordinateurs contenant des données importantes doivent être sauvegardés avant d'être utilisés dans nos sessions. Certains participants ont déjà commis des erreurs lors de travaux pratiques avec pour conséquence des pertes de données.
Supports	Copie électronique des présentations et travaux pratiques. Version électronique des données pour les travaux pratiques..

Matériel

La plateforme matérielle utilisée pendant les travaux pratiques de cette formation est la carte **BeagleBone Black**, dont voici les caractéristiques :

- 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.





Démonstrations

Les démos de cette formation utiliseront les périphériques matériels suivants:

- Une webcam USB
- Une carte d'extension d'écran tactile LCD connectée à la carte BeagleBone Black, pour afficher la vidéo capturée par la webcam.
- Nous utiliserons également une carte Arduino comme moyen pour mesurer précisément le temps de démarrage, pour montrer comment mettre en place des techniques de mesure matérielles.

1^{er} jour - Matin

Cours - Méthodes

- Comment mesurer le temps de démarrage
- Principales approches

TP - Construction du système

- Téléchargement du code source du chargeur de démarrage, du noyau et de Buildroot
- Prise en main de la carte, mise en place de la communication série
- Configuration de Buildroot et génération du système
- Configuration et compilation du chargeur de démarrage U-Boot. Préparation d'une carte SD pour démarrer le système.
- Configuration et compilation du noyau. Démarrage du système.



1^{er} Jour - Après-midi

Cours - Mesure du temps

- Techniques génériques par logiciel
- Techniques matérielles
- Solutions spécifiques à chaque étage du démarrage

TP - Mesure du temps - Solution logicielle

- Modification du système pour mesurer le temps au niveau des différentes étapes.
- Chronométrer les messages sur la console série
- Chronométrer le démarrage de l'application

TP - Mesure du temps - Solution matérielle

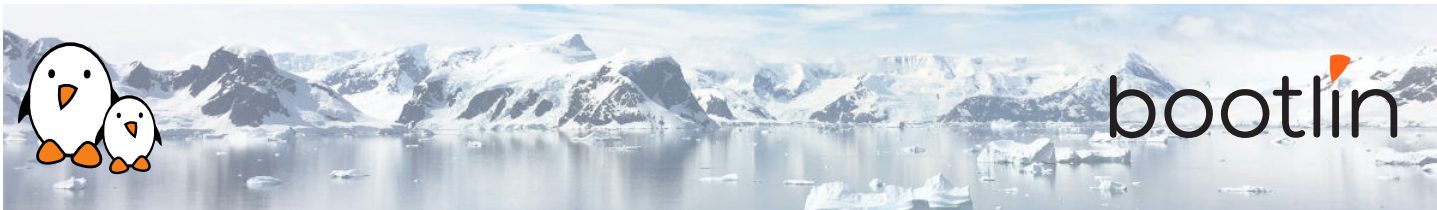
- Mesure du temps total de démarrage en positionnant une GPIO
- Mise en oeuvre d'une carte Arduino
- Préparation d'un circuit de test avec un afficheur à 7 segments
- Modification du DTS pour configurer les broches de la Bone Black en tant que GPIOs
- Modification de l'application pour piloter les GPIOs personnalisés

Cours - Optimisations des chaînes de compilation

- Introduction aux chaînes de compilation
- Bibliothèques C
- Informations de taille
- Mesure de la performance d'un exécutable avec la commande `time`

TP - Optimisations des chaînes de compilation

- Mesure du temps d'exécution de l'application
- Passage à une chaîne Thumb2
- Génération d'un SDK Buildroot pour recompiler plus vite



2^{ème} Jour - Matin

Cours - Optimisation de l'application

- Utilisation de `strace`
- Autres techniques de profiling

TP - Optimisation de l'application

- Rechercher d'options de configuration inutiles dans des applications
- Modification de ces options à travers Buildroot
- Expériences avec `strace` pour suivre l'exécution d'un programme

Cours - Optimisation du démarrage du système

- Utilisation de Bootchart
- Optimisation des scripts d'init
- Possibilité de démarrer directement votre application

TP - Optimisation du démarrage du système

- Utilisation de Buildroot pour supprimer scripts et commandes non nécessaires
- Une méthode pour identifier tous les fichiers inutilisés
- Simplification de BusyBox
- Démarrage de l'application en tant que programme init.

2^{ème} Jour - Après-midi

Cours - Optimisations de systèmes de fichiers

- Systèmes de fichiers disponibles, aspects de performance et de temps de démarrage
- Comment accélérer UBIFS
- Paramètres pour réduire le temps de démarrage
- Démarrer depuis un `initramfs`
- Utilisation d'exécutables statiques: contraintes de licence

TP - Optimisations de systèmes de fichiers

- Essayer et mesurer deux systèmes de fichiers bloc: `ext4` et `SquashFS`.
- Essai et benchmark de la solution `initramfs`. Contraintes en rapport avec cette solution.



Cours - Optimisations du noyau

- Utilisation d'*Initcall debug* to générer un *boot graph*
- Options de compression et liées à la taille
- Réduction ou suppression de la sortie console
- Plusieurs réglages pour réduire le temps de démarrage

TP - Optimisations du noyau

- Génération et analyse d'un *boot graph* pour le noyau
- Identifier et éliminer les fonctionnalités du noyau non nécessaires
- Trouver la meilleure option de compression pour votre système

3^{ème} Jour - Matin

TP - Optimisations du noyau

- Poursuite du TP

3^{ème} Jour - Après-midi

Cours - Optimisations du chargeur de démarrage

- Compiler U-Boot avec moins de fonctionnalités
- Réglages de configuration d'U-Boot qui ont un impact sur le temps de démarrage
- Optimisation du chargement du noyau
- Sauter le chargeur de démarrage - Comment modifier U-Boot pour activer son *Falcon mode*

TP - Optimisations du chargeur de démarrage

- Utilisation des techniques ci-dessus pour rendre le chargeur de démarrage le plus rapide possible
- Passer à un stockage plus rapide
- Sauter le chargeur de démarrage avec le *Falcon mode* d'U-Boot



Conclusion - Résultats obtenus

- Partage et comparaison des résultats obtenus par les différents groupes
- Questions / réponses, partage d'expérience avec le formateur