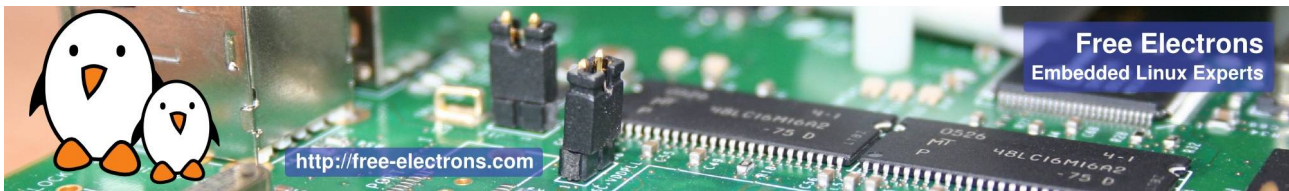


Formation « Développement de systèmes Linux embarqué »

Aperçu

Titre	Formation au développement de systèmes Linux embarqué
Aperçu	Chaînes de compilation croisée, bibliothèques standard C pour l'embarqué. Chargeurs de démarrage (bootloaders). Configuration et compilation du noyau Linux. Applications et bibliothèques légères pour systèmes embarqués. Systèmes de fichiers traditionnels et spécialisés pour stockage flash. Outils de développement de systèmes embarqués Linux. Développement et mise au point d'application sur le système embarqué. Contraintes temps-réel et Linux embarqué. Travaux pratiques avec une carte ARM.
Durée	5 jours. 40% de présentations et 60% de travaux pratiques.
Formateur	Thomas Petazzoni ou Michael Opdenacker (voir http://free-electrons.com/company/staff/)
Langue	Langue orale: Français Supports de cours: Anglais
Public ciblé	Ingénieurs développant des systèmes embarqués reposant sur Linux et des composants open-source.
Pré-requis	Connaissance et pratique de la ligne de commande Unix ou GNU/Linux Les personnes n'ayant pas cette expérience peuvent se former par elles-mêmes en utilisant notre support de formation disponible en ligne (http://free-electrons.com/docs/command-line/) Il est également possible d'avoir une journée supplémentaire de formation sur ce sujet.
Matériel nécessaire	<i>Dans le cadre des sessions publiques, l'ensemble du matériel est fourni par Free-Electrons</i> Vidéo-projecteur. Tableau blanc ou paper-board. Un ordinateur avec un lecteur de CD-ROM bootable sur chaque table (pour une ou deux personnes). L'ordinateur doit disposer d'au moins 512 Mo de RAM et une partition libre d'au moins 10 GB pour installer Linux (peut-être fait en avance ou durant la formation par les participants). L'utilisation de Linux dans une machine virtuelle n'est pas supportée , à cause de contraintes de connexion réelles avec le matériel. Si Linux est déjà installé, nous avons besoin d'une version 32 bits (i386) d'Ubuntu 9.04 (la variantes Xubuntu et Kubuntu conviennent aussi). Nous n'assurons pas de support pour les autres distributions, car nous ne pouvons pas tester toutes les versions de paquetages possibles. Connexion à Internet (directe ou par le proxy de la société). Les ordinateurs contenant des données précieuses doivent être sauvegardés avant d'être utilisés pour nos formations. Des personnes ont déjà commis des erreurs lors des formations et ont endommagé leurs données.
Supports	Version imprimée et électronique des supports de cours et des instructions des travaux pratiques.



Formation - Développement de système embarqué Linux

Nos supports de formation sont intégralement et gratuitement disponibles en ligne. Vous pouvez les consulter à l'adresse <http://free-electrons.com/doc/training/embedded-linux>. Ainsi, vous pouvez vérifier par vous-même si le contenu de la formation répond à vos besoins.

Matériel

Utilisation de cartes USB-A9263 de CALAO Systems dans la plupart des travaux pratiques

AT91SAM9263 ARM CPU d'ATMEL
 64 MB RAM, 256 MB flash
 2 USB 2.0 host
 1 USB device
 Port 100 Mbit Ethernet
 Alimenté par USB!
 Port série et JTAG par le port USB d'alimentation
 De nombreuses cartes d'extension disponibles



Jour 1 - Matin

TP - Mise en place de l'environnement de travail

En utilisant la distribution Xubuntu GNU/Linux (<http://xubuntu.org>)
 Installation de Xubuntu sur une partition libre du disque dur.

Cours - Introduction à Linux embarqué

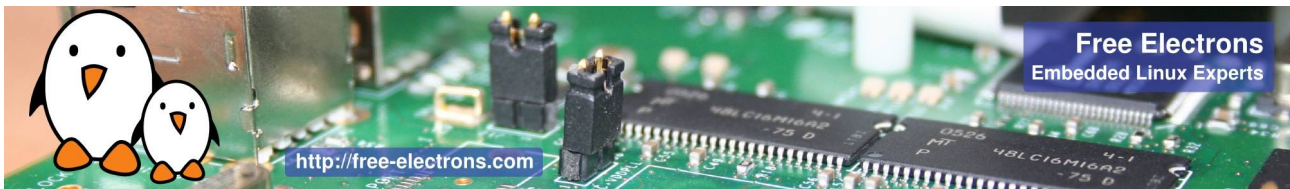
Avantages de Linux par rapport aux autres OS pour l'embarqué. Raisons pour choisir Linux.
 Aperçu général : comprendre l'architecture d'un système Linux embarqué. Aperçu des principaux éléments dans un système typique.
Le reste de la formation étudie chacun de ces éléments en détail.

Cours - Éléments d'administration système

Vous serez l'administrateur de votre système embarqué.
 Configurer le réseau.
 Manipuler les disques et partitions.
 Création et montage de systèmes de fichiers.
 Divers : propriétaire des fichiers, extinction, gestion des paquets, etc.

Cours - Chaîne de compilation croisée et bibliothèque standard C

Les composants d'une chaîne de compilation croisée.
 Choisir une bibliothèque standard C.
 Le contenu de la bibliothèque standard C.
 Les chaînes de compilation croisée prêtes à l'emploi.
 La construction automatisée d'une chaîne de compilation croisée.



Jour 1 - Après-midi

TP - Chaîne de compilation croisée	Cours - Chargeur de démarrage
<p>Configuration de Crosstool-NG Exécution pour construire une chaîne de compilation croisée personnalisée reposant sur la uClibc.</p>	<p>Chargeurs de démarrage existants Fonctionnalités des chargeurs de démarrage Installation d'un chargeur de démarrage Étude détaillée d'U-Boot</p>

TP - U-boot	Cours - Noyau Linux
<p>Configuration et compilation d'U-Boot à partir des sources. Installation sur la carte ARM Calao. Interaction au travers du port série.</p>	<p>Rôle et architecture générale du noyau Linux. Fonctionnalités disponibles dans le noyau Linux, en insistant sur les fonctionnalités utiles dans les systèmes embarqués. L'interface entre le noyau et les applications. Récupérer les sources. Comprendre les versions du noyau. Utilisation des patches.</p>

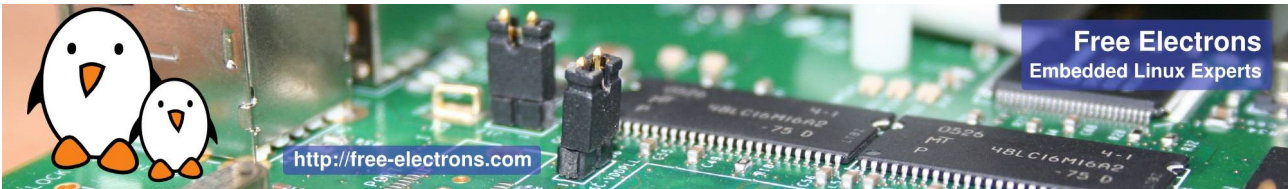
Jour 2 - Matin

TP - Sources du noyau	Cours - Configuration et compilation du noyau Linux
<p>Téléchargement des sources Application de patches</p>	<p>Configuration du noyau. Paramètres utiles pour les systèmes embarqués. Compilation native. Fichiers générés à l'issue de la compilation. Utilisation des modules noyau.</p>

TP - Configuration, compilation et démarrage du noyau	Cours - Démarrage
<p>Configuration, compilation et démarrage dans un système x86 virtuel.</p>	<p>Aperçu de la séquence de démarrage d'un système Linux. Paramètres du noyau. Démarrage au travers de NFS. Démarrage en utilisant un <i>initramfs</i>. Démarrage du système.</p>

Jour 2 - Après-midi

Cours - Compilation croisée du noyau
<p>Mise en place de la compilation croisée pour le noyau. Utilisation de configurations prêtes à l'emploi pour des cartes embarquées. Compilation croisée du noyau.</p>



TP - Compilation croisée du noyau et démarrage sur la carte

En utilisant la carte ARM Calao.

Configuration du noyau Linux et compilation croisée pour la carte ARM.

Mise en place d'un serveur TFTP sur la station de développement.

Téléchargement du noyau en utilisant le client TFTP d'U-Boot.

Démarrage du noyau depuis la RAM.

Copie du noyau vers la flash et démarrage depuis la flash.

Stockage des paramètres de démarrage en flash et automatisation du démarrage du noyau.

Cours - Busybox

Présentation de BusyBox. Intérêt pour les systèmes embarqués.

Configuration, compilation et installation.

Jour 3 - Matin

TP - Construction d'un minuscule système Linux embarqué avec BusyBox

Construction à partir de zéro d'un système de fichiers racine contenant un système Linux embarqué

Mise en place d'un noyau permettant de démarrer le système depuis un répertoire mis à disposition par la station de développement au travers de NFS.

Passage de paramètres au noyau pour le démarrage avec NFS.

Création complète du système de fichiers à partir de zéro : installation de BusyBox, création des fichiers spéciaux pour les périphériques.

Initialisation du système en utilisant le programme *init* de BusyBox.

Utilisation du serveur HTTP de BusyBox.

Contrôle de la cible à partir d'un navigateur Web sur la station de développement.

Mise en place des bibliothèques partagées sur la cible et développement d'une application d'exemple.

Jour 3 - Après-midi

Cours - Système de fichiers bloc

Systèmes de fichiers pour périphériques blocs.

Utilité des systèmes de fichiers journalisés.

Systèmes de fichiers en lecture seule.

Systèmes de fichiers en RAM.

Création de chacun de ces systèmes de fichiers.

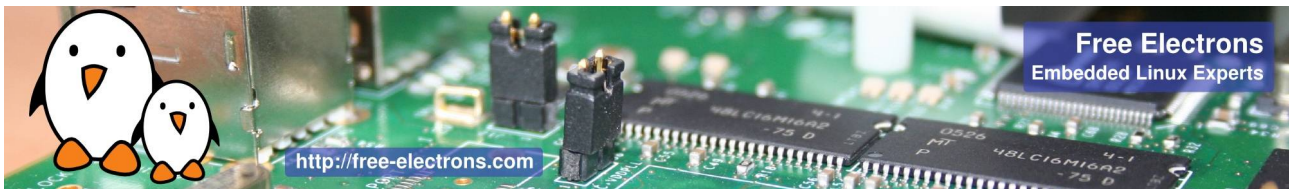
Suggestions pour les systèmes embarqués.

TP - Système de fichiers bloc

En utilisant la carte ARM Calao

Créer des partitions sur le stockage bloc.

Démarrage d'un système avec un assemblage de plusieurs systèmes de fichiers : SquashFS pour les applications, ext3 pour la configuration et les données utilisateur et tmpfs pour les fichiers temporaires.



Cours - Système de fichiers pour flash	TP - Systèmes de fichiers pour flash
<p>Le sous-système Memory Technology Devices du noyau Linux.</p> <p>Les systèmes de fichiers pour le stockage MTD : JFFS2, YAFFS2, UBIFS.</p> <p>Options de configuration du noyau.</p> <p>Partitions MTD.</p> <p>Montage d'images de systèmes de fichiers MTD.</p>	<p><i>Sur la carte ARM Calao</i></p> <p>Création de partitions sur la mémoire flash interne.</p> <p>Utilisation de SquashFS pour les applications.</p> <p>Utilisation de JFFS2 pour les données.</p>

Jour 4 - Matin

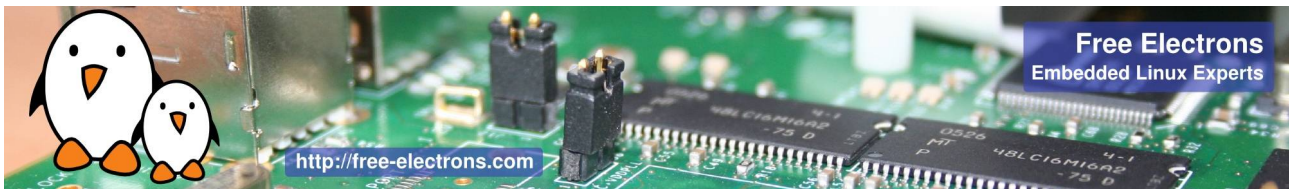
Cours - Réutilisation de composants open-source existants pour le système embarqué
<p>Motivations pour la réutilisation de composants existants.</p> <p>Trouver et choisir des composants libres et open-source existants.</p> <p>Les licences de Logiciels Libres et leurs conditions.</p> <p>Aperçu de composants typiquement utilisés dans les systèmes Linux embarqués : bibliothèques et systèmes graphiques (framebuffer, DirectFB, Gtk, Qt, etc.), utilitaires systèmes, bibliothèques et utilitaires réseau, bibliothèques multimédia, etc.</p> <p>Exemple d'un produit électronique de grande consommation reposant sur de nombreux composants open-source.</p> <p>Construction du système et intégration des composants.</p>

Cours - Compilation croisée de bibliothèques et d'applications	TP - Compilation croisée de bibliothèques et d'applications.
<p>Configuration, compilation croisée et installation de bibliothèques et d'applications pour un système embarqué</p> <p>Détails sur le système de compilation utilisé dans la plupart des composants open-source.</p> <p>Aperçu des principaux problèmes rencontrés lors de la réutilisation des composants.</p>	<p><i>Construction d'un système avec la bibliothèque graphique DirectFB</i></p> <p>Compilation et installation manuelle de plusieurs composants open-source.</p> <p>Apprentissage des principales techniques et des problèmes principaux.</p>

Jour 4 - Après-midi

TP - Compilation croisée de bibliothèques et d'applications
<p>... suite de la matinée</p>

Cours - Outils de construction de systèmes	TP - Construction d'un système avec Buildroot
<p>Outils de la communauté pour la construction automatisée de systèmes : Buildroot, Scratchbox et OpenEmbedded.</p> <p>Présentation détaillée de Buildroot.</p>	<p>Utilisation de Buildroot pour construire de façon automatisée un système similaire à celui du TP précédent.</p>



Jour 5 - Matin

Cours - Développement et débogage d'application

Langage de programmations et bibliothèques disponibles.
Aperçu de la bibliothèque C pour le développement d'applications.
Systèmes de construction pour votre application, comment utiliser des bibliothèques existantes dans votre application.
Environnements de développement intégrés (IDE) et lecteur de code source.
Débogueurs : débogage d'applications à distance avec gdb et gdbserver, analyse post-mortem d'une application.
Analyseurs de code, analyseurs mémoire, outils de profiling.
Développement depuis Windows.

TP - Développement et débogage d'application

Développement et compilation d'une application basée sur DirectFB
Utilisation de strace, ltrace et gdbserver pour déboguer une application de mauvaise qualité sur le système embarqué
Analyse post-mortem d'une application fautive.

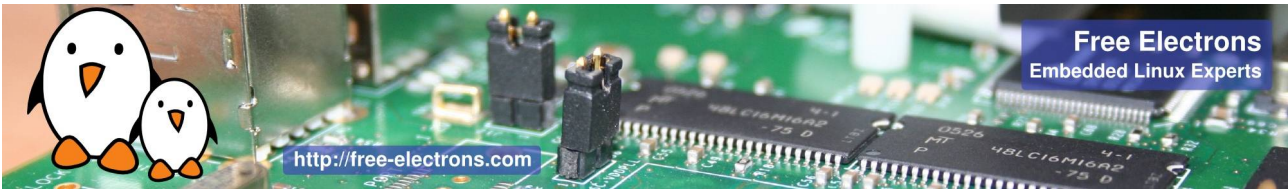
Jour 5 - Après-midi

Cours - Linux et le temps réel

Utile pour de nombreux types de systèmes, industriels ou multimédia.
Comprendre les sources de latence dans le noyau Linux standard.
Solutions temps réel mou : améliorations du noyau 2.6.
Comprendre et utiliser les patches RT preempt pour le noyau Linux.
Déboguage temps-réel du noyau. Mesure et analyse de la latence.
Xenomai, une solution temps réel dur pour Linux : fonctionnalités, concepts, implémentation et exemples.
Solutions temps-réel commerciales.

TP - Tests de latence Linux

Tests sur la carte ARM Calao
Développement d'une application reposant sur l'API temps-réel de Linux.
Mesure et comparaison de la latence d'ordonnancement entre un noyau Linux standard et un noyau RT preempt.



Sujets supplémentaires

Ces sujets peuvent être couverts à la fin de la formation si les autres thématiques ont été couvertes dans un temps plus court que prévu.

Cours - Connexion à chaud	TP - Connexion à chaud
<p>Udev : gestion des évènements matériels depuis l'espace utilisateur.</p> <p>Création et suppression des fichiers périphériques, identification des pilotes, notifications des programmes et des utilisateurs.</p> <p>Utilisation de mdev, une implémentation simple de udev.</p>	<p>Utilisation de mdev pour remplir le répertoire /dev/ avec les périphériques disponibles.</p> <p>Ajout de règles pour mdev, afin de monter automatiquement une clé USB lors de son insertion.</p>

Cours - Optimisations du système
<p>Idées, techniques et ressources pour réduire le coût, la taille, le temps de démarrage, le temps d'exécution et la consommation énergétique d'un système Linux embarqué.</p>