Implementing Linux on Embedded Systems

Objectives

- Understanding the architecture of the Linux system
- Learn how to install Linux on your hardware and create a BSP
- Explore the Linux system architecture
- Booting Linux
- Initializing the system
- Install existing packages on the target
- Learn how to install Linux on flash chips

All labs are conducted using the System Workbench for LinuxIDE.

Course environment

- Printed course material (in English)
- One Linux PC for two trainees.
- One target platform for two trainees

A version of Ac6 System Workbench for Linux Basic Edition is provided free of charge to each trainee

Prerequisite

- Good C programming skills
- Knowledge of Linux user programming (see our cours D0 - Programmation en mode utilisateur Linux)
- Preferably knowledge of Linux kernel and driver programming (see our cours D3 - Drivers Linux)

Environnement du cours

- Cours théorique
  - Support de cours au format PDF (en anglais) et une version imprimée lors des sessions en présentiel
  - Cours dispensé via le système de visioconférence Teams (si à distance)
  - Le formateur répond aux questions des stagiaires en direct pendant la formation et fournit une assistance technique et pédagogique
- Activités pratiques
  - Les activités pratiques représentent de 40% à 50% de la durée du cours
  - Elles permettent de valider ou compléter les connaissances acquises pendant le cours théorique.
  - Exemples de code, exercices et solutions
  - Pour les formations à distance:
    - Un PC Linux en ligne par stagiaire pour les activités pratiques, avec tous les logiciels nécessaires préinstallés.
    - Le formateur a accès aux PC en ligne des stagiaires pour l'assistance technique et pédagogique
    - Certains travaux pratiques peuvent être réalisés entre les sessions et sont vérifiés par le formateur lors de la session suivante.
  - Pour les formations en présentiel::
    - Un PC (Linux ou Windows) pour les activités pratiques avec, si approprié, une carte cible embarquée.
    - Un PC par binôme de stagiaires s'il y a plus de 6 stagiaires.
  - Pour les formations sur site:
    - Un manuel d'installation est fourni pour permettre de préinstaller les logiciels nécessaires.

    - Le formateur vient avec les cartes cible nécessaires (et les remporte à la fin de la formation).
- Une machine virtuelle préconfigurée téléchargeable pour refaire les activités pratiques après le cours
- Au début de chaque session (demi-journée en présentiel) une période est réservée à une interaction avec les stagiaires pour s'assurer que le cours répond à leurs attentes et l'adapter si nécessaire

Audience visée

- Tout ingénieur ou technicien en systèmes embarqués possédant les prérequis ci-dessus.

Modalités d'évaluation

- Les prérequis indiqués ci-dessus sont évalués avant la formation par l'encadrement technique du stagiaire dans son entreprise, ou par le stagiaire lui-même dans le cas exceptionnel d'un stagiaire individuel.
- Les progrès des stagiaires sont évalués de deux façons différentes, suivant le cours:
    - Pour les cours se prêtant à des exercices pratiques, les résultats des exercices sont vérifiés par le formateur, qui aide si nécessaire les stagiaires à les réaliser en apportant des précisions supplémentaires.
    - Des quizz sont proposés en fin des sections ne comportant pas d'exercices pratiques pour vérifier que les stagiaires ont assimilé les points présentés
- En fin de formation, chaque stagiaire reçoit une attestation et un certificat attestant qu'il a suivi le cours avec succès.
    - En cas de problème dû à un manque de prérequis de la part du stagiaire, constaté lors de la formation, une formation différente ou complémentaire lui est proposée, en général pour conforter ses prérequis, en accord avec son responsable en entreprise le cas échéant.

Plan

First Day

Introduction to Linux

- Linux history and Version management
- Linux system architecture
    - Processes and MMU
    - System calls
    - Shared libraries
- Linux components
    - Toolchain
    - Bootloader
    - Kernel
    - Root file system
- Linux distributions
- Linux packages
- The various licenses used by Linux (GPL, LGPL, etc)

Linux tools for embedded systems

- Boot loaders (UBoot, Redboot, barebox)
- Optimized libraries (eglibc, uClibc)
- Toolchains
- Embedded GUIs
- Busybox
- Embedded distributions
    - Commercial
    - Standard
    - Tools for building custom distributions

Introduction to System Workbench

- Overview
  - Eclipse
  - Kernel and modules
  - Platforms and Root file-systems
- The build system architecture
  - Building individual packages
  - Building platforms
  - Building Root file-systems

Exercise : *Building a root file system a pre-defined platform template*

Developing applications with System Workbench

- Creating a Linux program
- Creating a library
  - Static library
  - Shared library
- Debugging on the target
  - Using an SSH connection
  - Debugging shared libraries

Exercise : *Create a small program, with a custom shared library, and debug it on the target*

Using U-Boot

- Introduction to U-Boot
- Booting the board through U-Boot
  - Booting from NOR
  - Booting from NAND
  - Booting from eMMC
- U-Boot environment variables
  - User-defined variables
  - Predefined variables
  - Variables substitution
- The U-Boot minimal shell
  - Writing scripts in variables
  - Executing scripts
  - Using variables in scripts: the set-script pattern
- U-Boot main commands
  - Booting an OS
  - Accessing flash chips
  - Accessing file systems (NFS, FAT, EXTx, JFFS2&)
- The full U-Boot shell
  - Script structure
  - Control flow instructions (if, for&)
- Booting Linux
  - Linux kernel parameters
  - The Linux startup sequence

Exercise : *Writing a script to configure the network and pass this configuration to the Linux kernel*
Exercise : *Booting the board on NFS, using pre-existing images*
Exercise : *Writing scripts to choose between boot from flash or from the network*

Second Day

Building U-Boot

- Building and installing U-Boot with its native build system
- Building U-boot with System Workbench

Exercise : *Configuring and building u-boot with its native build system*
Exercise : *Building u-boot from System Workbench*

Building the kernel

- The Linux build system
  - Downloading stable source code
- Getting a tarball
- Using GIT
  - Configuring the kernel
  - Compiling the kernel and its modules
- Modules delivered in-tree
- Out-of-tree modules
  - Installing the kernel and the modules

Exercise : *Configuring and compiling a target kernel for the target board with the kernel build system*

- Kernel projects
  - Creating a kernel project
  - Selecting the architecture and configuration
  - Customizing the configuration
  - Compiling the kernel

Exercise : *Configure and compile the kernel in the platform*

- Module projects
  - Creating a module project
  - Linking it to a kernel project
  - Creating and building modules

Exercise : *Add and configure an external module*

Exercise : *Exercise: Configuring and compiling a target kernel for the target board with System Workbench*

Building packages

- Packages
  - Tools to build packages (gcc, Makefile, pkg-config&)
  - Autotools
  - Cross-compiling a package with autotools
- The all-in-one applications
  - Busybox, the basic utilities
  - Dropbear: encrypted communications (ssh)
- Automatically starting a program at boot
  - Initialization systems (busybox init, system V init)

Exercise : *Cross-compiling an autotools-based package*

Creating a Linux Platform

- Creating a platform project
  - Importing a pre-configured platform
  - Creating a platform from scratch
- Configuring the platform
  - Source and installation directories
  - Link to a target Rootfs
  - Build configurations

Exercise : *Create and configure a minimum platform from scratch, using library packages*

- Populating the build environment
  - Import packages in the build environment
  - Build individual packages
  - Build the whole platform

Exercise : *Build the platform, manually building some packages*

- Adding packages to a platform
  - From a library
  - From an existing Eclipse project

Exercise : *Add the previously developed application to the platform*

- Creating a new package
  - Specifying the source

- Patching the official sources
- Adding package-specific resources
- Adding package configuration directives

Exercise : *Add a new open-source package to the platform*

Third Day

Embedded file systems

- Storage interfaces
  - Block devices
  - MTD
- Flash memories and Linux MTDs
  - NOR flashes
  - NAND flashes
  - ONENAND flashes
- The various flash file system formats
  - JFFS2, YAFFS2, UBIFS
- Read-only file system
  - CRAMFS, SQUASHFS
- Standard Linux file systems
  - Ext2/3/4, FAT, NFS
  - Ramdisks and initrd
  - Creating an initramfs
  - Booting through an initramfs
- Choosing the right file system formats
- Flashing the file system

Creating a root file system

- Manually building your root file system
  - Device nodes, programs and libraries
  - Configuration files (network, udev, &)
  - Installing modules
  - Looking for and installing the needed libraries
  - Testing file system consistency and completeness
- Package management
  - ipkg

Exercise : *Manually creating a minimal root file system using busybox and dropbear*

- Creating a rootfs project
  - Creating the rootfs structure
  - Add files to the base structure
- Edit standard configuration files
  - File systems
  - Initialization
  - Starting applications

Renseignements pratiques

Renseignements : 3 jours