Linux embedded and Yocto Project training (3 days - combo training)

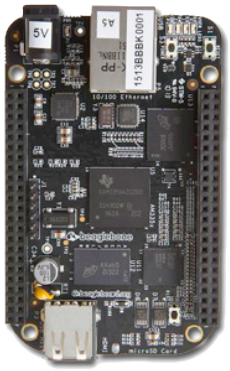
Title	Linux embedded and Yocto Project training
Overview	Understanding the Linux filesystem Setup TFTP and NFS services Understanding bootloaders u-boot bootloader Understanding the Linux kernel Configuring and building the Linux kernel Linux Device Tree OpenEmbedded and Yocto Project overview Using it to build a root filesystem and run it on your target Writing and extending recipes Creating layers Practical labs with ARM-based board
Duration	**THREE** day - 24 hours (8 hours per day). 50% of lectures, 50% of practical labs (approx.)
Trainer	Marco Cavallini m.cavallini (AT) koansoftware.com
Language	Oral lectures: English or Italian Materials: English.
Audience	People that need to learn how to use a Linux system People using and managing an embedded Linux system People that need to learn how to configure and build a whole Linux system using Yocto Project People developing Linux kernel and user-space applications People creating Yocto Project recipes and layers
Prerequisites	**Knowledge of Linux commands** as covered in our embedded Linux training (`http://koansoftware.com/en/content/linux-embedded-course`) **Knowledge and practice of Unix or GNU/Linux commands** People lacking experience on this topic should not attend this course.

Required equipment	**For public sessions** Everything is supplied by KOAN in public sessions except the PC. Participants must have **their own PC laptop computer** with: • PC computers with at least **2GB of RAM**, and **40GB** of free disk space. • **VirtualBox 5** installed. • We will work with Lubuntu Desktop 14.04 (64 bit) We don't support other distributions, because we can't test all possible package versions. • **Connection to the Internet** (direct or through the company proxy). • **PC computers with valuable data must be backed up** before being used in our sessions. Some people have already made mistakes during our sessions and damaged work data. **For on-site sessions** please add the following • Video projector • **Connection to the Internet** (direct or through the company proxy).
Materials	Print and electronic copies of presentations and labs. Electronic copy of lab files.

Hardware

The hardware platform used for the practical labs of this training session is the **BeagleBone Black** board, which features:

- An ARM AM335x processor from Texas Instruments (Cortex-A8 based), 3D acceleration, etc.
- 512 MB of RAM
- 4 GB of on-board eMMC storage (4 GB in Rev C)
- USB host and device
- HDMI output
- 2 x 46 pins headers, to access UARTs, SPI buses, I2C buses



Note:
Content and order of this agenda may slightly vary between sessions and will be determined by the participants and the specific needs of the class.

Day 1 - Morning

Lecture - Setup and Introduction

- Virtual machine setup
- Introduction to embedded linux
- Advantages of using linux
- Systems running linux
- Typical embedded hardware
- System architecture

Lecture - Linux commands and filesystem

- Linux commands
- Linux filesystem
- Virtual filesystems
- Understanding the development process

Lab - Using linux

Using the Virtual Machine

- Using the Unix command line
- using the vi text editor
- Discovering procfs and sysfs
- Using TFTP and NFS connect the board to develompent PC

Day 1 - Afternoon

Lecture - Configuring, compiling and booting the Linux kernel

- Busybox
- Bootloaders
- u-boot
- Autotools concepts
- Booting the kernel using TFTP and NFS

Lab - Kernel configuration, cross-compiling and booting on NFS

Using the Virtual Machine

- Configuring TFTP server on the host machine
- Configuring NFS server on the host machine
- Flash a Linux image on a SDCard
- Booting the target board using TFTP and NFS

Day 2 - Morning

Lecture - Linux kernel

- Linux kernel sources structure
- Details about the API provided to kernel drivers
- Cross compilator toolchains
- Cross-compiling the kernel for the target
- Device Tree

Lecture - Linux kernel details

- Linux kernel introduction
- Linux versioning
- Generating patches with diff
- Understanding the kernel development process

Lab - Using linux

Using the Virtual Machine

- Extracting a generic linux kernel
- Applying patches to the kernel with patch
- Configuring the kernel

Day 2 - Afternoon

Lecture - Configuring, compiling and booting the Linux kernel

- Linux kernel configuration
- Kernel booting parameters
- Native and cross-compilation generated files
- CPU pin muxing
- Device Tree
- The init process

Lab - Kernel configuration, cross-compiling and booting on NFS

Using the Virtual Machine

- Cross compile a customized kernel
- Run a modified Linux image on your target board
- Play around with Embedded Linux on your board

Day 3 - Morning

Lecture - Yocto Project introduction

- Yocto Project overview
- How to setup the Yocto Project build system
- Organization of the project source tree
- Building a root filesystem image using the Yocto Project

Lecture - OpenEmbedded and Yocto Project

- General concepts of a build system
- Origin of Yocto Project
- Yocto Project recipes
- Yocto Project meta layers
- Configuring the build system
- Customizing the package selection

Lab - Running Yocto on the host

Using the Virtual Machine

- Setup the Poky reference build system
- Building a system image
- Creating a meta layer with Yocto Project
- Creating an example recipe with Yocto Project

Day 3 - Afternoon

Lecture - Yocto Project

- Writing a minimal recipe
- Adding dependencies
- Development workflow with *bitbake*
- Meta layers customization

Lab - Running linux on the target

Using the ARM board

- Create a custom recipe for a new package *nInvaders*
- Flash a new Linux image on a SDCard
- Writing a recipe for *nInvaders*
- Adding *nInvaders* to the final image
- Play around with generated image on your board