

## Linux embedded and Yocto Project training (2 days - combo training)



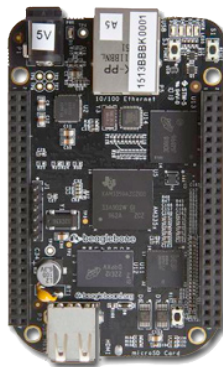
<b>Title</b>	Linux embedded and Yocto Project training
<b>Overview</b>	<ul style="list-style-type: none"> <li>Understanding bootloaders</li> <li>u-boot bootloader</li> <li>Understanding the Linux kernel</li> <li>Configuring the Linux kernel</li> <li>Building the Linux kernel</li> <li>Linux Device Tree</li> <li>OpenEmbedded and Yocto Project overview</li> <li>Using it to build a root filesystem and run it on your target</li> <li>Writing and extending recipes</li> <li>Creating layers</li> <li>Practical labs with ARM-based board</li> </ul>
<b>Duration</b>	<b>TWO</b> day - 16 hours (8 hours per day). 50% of lectures, 50% of practical labs (approx.)
<b>Trainer</b>	Marco Cavallini m.cavallini (AT) koansoftware.com
<b>Language</b>	Oral lectures: English or Italian Materials: English.
<b>Audience</b>	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.
<b>Prerequisites</b>	<b>Knowledge of embedded Linux</b> as covered in our Linux embedded training (LEVEL 1) ( <a href="http://koansoftware.com/en/content/linux-embedded-course">http://koansoftware.com/en/content/linux-embedded-course</a> )  <b>Knowledge and practice of Unix or GNU/Linux commands</b> <b>Knowledge of TFTP and NFS</b> People lacking experience on this topic should not attend this course.

<b>Required equipment</b>	<p><b>For public sessions</b> Everything is supplied by KOAN in public sessions except the PC. Participants must have <b>their own PC laptop computer</b> with:</p> <ul style="list-style-type: none"> <li>• PC computers with at least <b>2GB of RAM</b>, and <b>40GB</b> of free disk space.</li> <li>• <b>VirtualBox 5</b> installed.</li> <li>• We will work with <b>Lubuntu Desktop 14.04 (64 bit)</b> We don't support other distributions, because we can't test all possible package versions.</li> <li>• <b>Connection to the Internet</b> (direct or through the company proxy).</li> <li>• <b>PC computers with valuable data must be backed up</b> before being used in our sessions. Some people have already made mistakes during our sessions and damaged work data.</li> </ul> <p><b>For on-site sessions</b> please add the following</p> <ul style="list-style-type: none"> <li>• Video projector</li> <li>• <b>Connection to the Internet</b> (direct or through the company proxy).</li> </ul>
<b>Materials</b>	<p>Print and electronic copies of presentations and labs. Electronic copy of lab files.</p>

## 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 - Linux kernel

- Linux kernel sources structure
- Details about the API provided to kernel drivers
- Cross compiler 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
- Busybox
- Bootloaders
- u-boot

### Lab - Using linux

#### *Using the Virtual Machine*

- Extracting a generic linux kernel
- Applying patches to the kernel with patch
- Configuring the kernel
- 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 1 - 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 2 - 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 2 - 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