Embedded Linux development training

3 days session

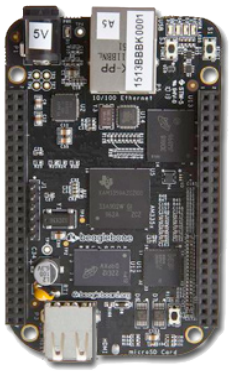
Title	Embedded Linux development training
Overview	Understanding the Linux kernel Building the Linux kernel Developing with Yocto Project Developing Linux device drivers Linux application debugging Working with the kernel development community Practical labs with ARM-based board
Duration	**three** days - 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 developing devices using the Linux kernel People supporting embedded Linux system developers.
Prerequisites	**Knowledge of embedded Linux** 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: • USB3 port support (for Disk provided) We will use a 32GB USB3 disk to work with Lubuntu 12.04 (32 bit) • USB to power the target board • USB to connect the serial adapter (provided) • Ethernet connector (for communication with the target) • Wifi • **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: The order and the content of the following program may vary slightly

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 Cross compilation

- Toolchain components
- Understanding the development process
- C libraries
- Toolchain options

Lab - Using linux

Using the Virtual Machine

- Using the Unix command line
- Using the vi text editor
- Using the apt package manager
- Discovering procfs and sysfs

Day 1 - Afternoon

Lecture - Configuring, compiling and booting the Linux kernel

- Embedded linux development environments
- Linux kernel features
- Linux versioning schemes

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

Using the Virtual Machine

- Get the kernel sources from the official location
- Check the authenticity of the kernel sources

Day 2 - Morning

Lecture - Yocto Project introduction

- Yocto Project overview
- How to setup the Yocto Project build system

Lecture - Yocto Project

- Yocto Project meta layers
- Yocto Project recipes

Lab - Running Yocto on the host

Using the Virtual Machine

- Setup a Yocto Project build system
- Creating a meta layer with Yocto Project
- Creating a recipe with Yocto Project

Day 2 - Afternoon

Lecture - Linux kernel and device drivers

- Linux kernel configuration
- Kernel booting parameters.
- Booting the kernel using NFS.
- Native and cross-compilation generated files.

Lab - Running linux on the target

Using the ARM board

- Configure the TFTP and the NFS server
- Flash a Linux image on a SDCard
- Launch the Linux image on your target board
- Play around with Embedded Linux on your board

Day 3 - Morning

Lecture - Kernel init and Bootloaders

- Cross-compiling the kernel for the target
- Linux kernel sources structure
- Linux driver development
- Details about the API provided to kernel drivers

Lecture - Linux filesystems - Busybox

- Kernel initialization
- Bootloaders
- Boot sequence
- u-boot
- Linux root filesystem

Lab - Device driver

Using the ARM board

- Creating a basic device driver
- Creating a simple character driver

Day 3 - Afternoon

Lecture - Application debugging

- Block filesystems
- Flash filesystems
- Virtual filesystems
- Busybox

Lab - Running linux on the target

Using the ARM board

- Debugging user space applications
- Remote debugging user space applications