



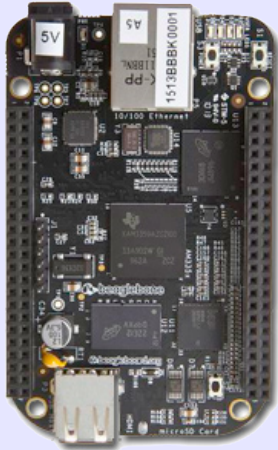

# Yocto Project and OpenEmbedded training

## 3-day session

<b>Title</b>	<b>Yocto Project and OpenEmbedded development training</b>
<b>Overview</b>	Understanding the Yocto Project Using it to build a root filesystem and run it on your target Writing and extending recipes Creating layers Integrating your board in a BSP Creating custom images Application development with the Yocto Project SDK
<b>Duration</b>	<b>Three</b> days - 24 hours (8 hours per day). 40% of lectures, 60% of practical labs.
<b>Trainer</b>	One of the engineers listed on <a href="https://bootlin.com/training/trainers/">https://bootlin.com/training/trainers/</a>
<b>Language</b>	Oral lectures: English, French. Materials: English.
<b>Audience</b>	Companies and engineers interested in using the Yocto Project to build their embedded Linux system.
<b>Prerequisites</b>	<b>Knowledge of embedded Linux</b> as covered in our embedded Linux training ( <a href="https://bootlin.com/training/embedded-linux/">https://bootlin.com/training/embedded-linux/</a> ) <b>Knowledge and practice of Unix or GNU/Linux commands</b> People lacking experience on this topic should get trained by themselves, for example with our freely available on-line slides: <a href="https://bootlin.com/blog/command-line/">https://bootlin.com/blog/command-line/</a>



<b>Required equipment</b>	<p><b>For on-site sessions only.</b> Everything is supplied by Bootlin in public sessions.</p> <ul style="list-style-type: none"><li>• Video projector</li><li>• PC computers with at least 8 GB of RAM, a CPU at least equivalent to an Intel Core i5 and Ubuntu Linux installed in a <b>free partition of at least 40 GB. Using Linux in a virtual machine is not supported</b>, because of issues connecting to real hardware.</li><li>• We need Ubuntu Desktop 16.04 (Xubuntu and other variants are fine). We don't support other distributions, because we can't test all possible package versions.</li><li>• <b>High Speed 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>
<b>Materials</b>	Electronic copies of presentations and labs. Electronic copy of lab files.

<b>Hardware, first option</b>	<b>Hardware, second option</b>
<p>BeagleBone Black board</p> <ul style="list-style-type: none"><li>• An ARM AM335x processor from Texas Instruments (Cortex-A8 based), 3D acceleration, etc.</li><li>• 512 MB of RAM</li><li>• 2 GB of on-board eMMC storage (4 GB in Rev C)</li><li>• USB host and device</li><li>• HDMI output</li><li>• 2 x 46 pins headers, to access UARTs, SPI buses, I2C buses and more.</li></ul> 	<p>STMicroelectronics STM32MP157A-DK1 Discovery board</p> <ul style="list-style-type: none"><li>• STM32MP157A (dual Cortex-A7) CPU from STMicroelectronics</li><li>• USB powered</li><li>• 512 MB DDR3L RAM</li><li>• Gigabit Ethernet port</li><li>• 4 USB 2.0 host ports</li><li>• 1 USB-C OTG port</li><li>• 1 Micro SD slot</li><li>• On-board ST-LINK/V2-1 debugger</li><li>• Arduino Uno v3-compatible headers</li><li>• Audio codec</li><li>• Misc: buttons, LEDs</li></ul> 



## Day 1 - Morning

---

### Lecture - Introduction to embedded Linux build systems

- Overview of an embedded Linux system architecture
- Methods to build a root filesystem image
- Usefulness of build systems

### Lecture - Overview of the Yocto Project and the Poky reference system

- Organization of the project source tree
- Building a root filesystem image using the Yocto Project

### Lab - First Yocto Project build

- Downloading the Poky reference build system
- Building a system image

## Day 1 - Afternoon

---

### Lecture - Using Yocto Project - basics

- Organization of the build output
- Flashing and installing the system image

### Lab - Flashing and booting

- Flashing and booting the image on the board

### Lecture - Using Yocto Project - advanced usage

- Configuring the build system
- Customizing the package selection

### Lab - Using NFS and configuring the build

- Configuring the board to boot over NFS
- Learn how to use the PREFERRED\_PROVIDER mechanism



## Day 2 - Morning

---

### Lecture - Writing recipes - basics

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

### Lab - Adding an application to the build

- Writing a recipe for *nInvaders*
- Adding *nInvaders* to the final image

### Lecture - Writing recipes - advanced features

- Extending and overriding recipes
- Adding steps to the build process
- Learn about classes
- Analysis of examples
- Logging
- Debugging dependencies

## Day 2 - Afternoon

---

### Lab - Learning how to configure packages

- Extending a recipe to add configuration files
- Using `ROOTFS_POSTPROCESS_COMMAND` to modify the final rootfs
- Studying package dependencies

### Lecture - Layers

- What layers are
- Where to find layers
- Creating a layer

### Lab - Writing a layer

- Learn how to write a layer
- Add the layer to the build
- Move *nInvaders* to the new layer



## Day 3 - Morning

---

### Lecture - Writing a BSP

- Extending an existing BSP
- Adding a new machine
- Bootloaders
- Linux and the linux-yocto recipe
- Adding a custom image type

### Lab - Implementing the kernel changes

- Extend the kernel recipe to add the nunchuk driver
- Configure the kernel to compile the nunchuk driver
- Play *nInvaders*

## Day 3 - Afternoon

---

### Lecture - Creating a custom image

- Writing an image recipe
- Adding users/groups
- Adding custom configuration
- Writing and using package groups recipes

### Lab - Creating a custom image

- Writing a custom image recipe
- Adding *nInvaders* to the custom image

### Lecture - Creating and using an SDK

- Understanding the purpose of an SDK for the application developer
- Building an SDK for the custom image

### Lab - Experimenting with the SDK

- Building an SDK
- Using the Yocto Project SDK