Embedded Linux development with Buildroot training

3-day session

Title	**Embedded Linux development with Buildroot training**
Overview	Introduction to Buildroot Managing and building the configuration Buildroot source and build trees Toolchains in Buildroot Managing the Linux kernel configuration Root filesystem Download infrastructure GNU Make 101 Integrating new packages Advanced package aspects Analyzing the build Advanced topics Application development with Buildroot Understanding the Buildroot internals Buildroot community: support and contribution What's new in Buildroot?
Materials	Check that the course contents correspond to your needs: `https://bootlin.com/doc/training/buildroot`.
Duration	**Three** days - 24 hours (8 hours per day). 40% of lectures, 60% of practical labs.
Trainer	**Thomas Petazzoni**. Thomas is a major Buildroot developer since 2009, with more than 2700 patches integrated and an active participation to the development process.
Language	Oral lectures: English, French. Materials: English.
Audience	Companies already using or interested in using Buildroot to build their embedded Linux systems.

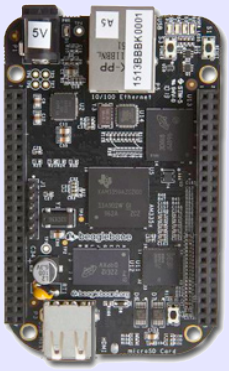
Prerequisites	**Knowledge of embedded Linux** as covered in our embedded Linux course:
	`https://bootlin.com/training/embedded-linux/`
	
	Knowledge and practice of UNIX or GNU/Linux commands
	People lacking experience on this topic should get trained by themselves, for example with our freely available on-line slides:
	`https://bootlin.com/blog/command-line/`

Required equipment	**For on-site sessions only.** Everything is supplied by Bootlin in public sessions. • Video projector • PC computers with at least 8 GB of RAM, and Ubuntu Linux installed in a **free partition of at least 30 GB. Using Linux in a virtual machine is not supported**, because of issues connecting to real hardware. • 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. • **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.
Materials	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**, which features:

- An ARM AM335x processor from Texas Instruments (Cortex-A8 based), 3D acceleration, etc.
- 512 MB of RAM
- 2 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 and more.



Day 1 - Morning

Lecture - Embedded Linux and build system introduction

- The general architecture of an embedded Linux system
- Build systems vs. binary distributions
- Role of a build system
- Comparison of existing build systems

Lecture - Introduction to Buildroot

- Key facts about the project
- Getting Buildroot
- Basic configuration of Buildroot
- Doing a first build

Lab - Basic Buildroot usage

- Getting and setting up Buildroot
- Configuring and building a basic system with Buildroot for the BeagleBone Black
- Flash and test the generated system on the BeagleBone Black

Lecture - Managing the build and configuration

- Out of tree build
- Using and creating *defconfigs*
- Defconfig fragments
- Other building tips

Day 1 - Afternoon

Lecture - Buildroot source and build trees

- Details about the Buildroot source code organization
- Details about the Buildroot build tree

Lecture - Toolchains in Buildroot

- The different choices for using toolchains in Buildroot
- Overview of the toolchain options
- Using existing binary toolchains, such as Bootlin toolchains, understanding *multilib* capabilities and integration of toolchains in Buildroot
- Generating custom toolchains with *Crosstool-NG*, and re-use them as external toolchains

Lecture - Managing the Linux kernel configuration

- Loading, changing and saving the kernel configuration

Lecture - Root filesystem construction in Buildroot

- Understand how Buildroot builds the root filesystem: *skeleton*, installation of packages, overlays, *post-build* and *post-image* scripts.
- Customization of the root filesystem contents
- System configuration: *console* selection, various `/dev` management methods, the different `init` implementations, etc.
- Understand how Buildroot generates filesystem images

Lab - Root filesystem customization

- Explore the build output
- Customize the root filesystem using a *rootfs overlay*
- Customize the kernel with patches and additional configuration options
- Add more packages
- Use *defconfig* files and *out of tree* build

Day 2 - Morning

Lecture - Download infrastructure in Buildroot

- Downloading logic
- Primary site and backup site, doing offline builds
- VCS download, integrity checking
- Download-related *make* targets

Lecture - GNU Make 101

- Basics of make rules
- Defining and referencing variables
- Conditions, functions
- Writing recipes

Lecture - Integrating new packages in Buildroot	Lab - New packages in Buildroot
• How to integrate new packages in the Buildroot configuration system • Understand the different package infrastructures: for *generic*, *autotools*, *CMake*, *Python* packages and more. • Writing a package `Config.in` file: how to express dependencies on other packages, on toolchain options, etc. • Details on writing a package recipe: describing the package source code location, download method, configuration, build and installation steps, handling dependencies, etc.	• Create a new package for *nInvaders* • Understand how to add dependencies • Add patches to *nInvaders* for *Nunchuk* support

Day 2 - Afternoon

Lecture - Advanced package aspects	Lab - Advanced packages
• Licensing report • Patching support: patch ordering and format, global patch directory, etc. • User, permission, device tables • Init scripts and systemd unit files • Config scripts • Understanding *hooks* • Overriding commands • Legacy handling • Virtual packages	• Package an application with a mandatory dependency and an optional dependency • Package a library, hosted on GitHub • Use *hooks* to tweak packages • Add a patch to a package

Day 3 - Morning

Lecture - Analyzing the build: licensing, dependencies, build time

- Usage of the legal information infrastructure
- Graphing dependencies of packages
- Collecting and graphing build time information

Lecture - Advanced topics

- `BR2_EXTERNAL` to store customizations outside of the Buildroot sources
- Package-specific targets
- Understanding rebuilds
- Tips for building faster

Lab - Advanced aspects

- Use build time graphing capabilities
- Use dependency graphing capabilities
- Use licensing report generation, and add licensing information to your own packages
- Use `BR2_EXTERNAL`

Day 3 - Afternoon

Lecture - Application development with Buildroot

- Using Buildroot during application development
- Usage of the Buildroot environment to build applications outside of Buildroot
- Generate an SDK for other developers
- Remote debugging with Buildroot

Lab - Application development with Buildroot

- Build and run your own application
- Remote debug your application
- Use `<pkg>_OVERRIDE_SRCDIR`

Lecture - Understanding Buildroot internals

- Detailed description of the Buildroot build process: toolchain, packages, root filesystem construction, stamp files, etc.
- Understanding virtual packages.

Lecture - Getting support and contributing, what's new in Buildroot

- Getting support: *Bugzilla*, *mailing list*, *IRC*
- Contributing: understanding the development process, how to submit patches
- What's new in Buildroot: summary of the major changes since the last two years