Embedded Linux Development with Yocto Project

Otavio Salvador
Daiane Angolini

Chapter No. 1
"Meeting the Yocto Project"
In this package, you will find:
A Biography of the authors of the book
A preview chapter from the book, Chapter NO.1 "Meeting the Yocto Project"
A synopsis of the book’s content
Information on where to buy this book

About the Authors

Otavio Salvador loves to play video games and started his free software activities in 1999. In 2002, he founded O.S. Systems, a company focused on embedded system development services and consultancy worldwide, creating and maintaining customized BSPs and helping companies with their release management challenges. This resulted in him joining the OpenEmbedded community in 2008, when he became an active contributor to the OpenEmbedded project, culminating in his attribution as the maintainer of the Freescale ARM BSP layer in the Yocto Project in 2011.

For More Information:
**Daiane Angolini** has been focusing on embedded technologies for the past 8 years. Since 2008, she has been working on Freescale Semiconductors as an application engineer, on internal development and porting custom applications from Android to Freescale architectures, and on customer support for ARM processors of the i.MX family, while also participating in Freescale forums. She has been working with the Yocto Project tools through meta-fsl-arm, the BSP meta layer that provides board support for Freescale ARM machines, since 2012. The desire to become an expert in ice cream making has been keeping her busy in her spare time for the past year.

We initially want to thank our families. They provided lovely support and helped us to get on track for this project.

This project has only been possible because we had support from many people who provided insights, reviews, material, and guidance during the full period of conception and production of this book. We'd like to give special thanks to (in alphabetic order): Alex González, Alexandru Vaduva, Harsha Bharwani, Jeffrey Osier-Mixon, John Weber, Manan Badani, Paul Eggleton, Rogerio Nunes, Radek Dostál, Sageer Parkar, and Sankalp Pawar

- Otavio Salvador and Daiane Angolini
Embedded Linux Development with Yocto Project

Considering the current technology trend, Linux is the next big thing. Linux has consistently released cutting-edge open source products, and embedded systems have been added to the technological portfolio of mankind.

The Yocto Project is in an optimal position to be the choice for your projects; it provides a rich set of tools to help you to use most of your energy and resources in your product development, instead of reinventing the wheel.

The usual tasks and requirements for embedded Linux-based products and development teams were the guidelines for this book's conception. Written by active community members with a practical and straightforward approach, it is a stepping stone for both your learning curve and your product's project.

What This Book Covers

Chapter 1, Meeting the Yocto Project, presents the history of the Yocto Project, showing the parts that compose it.

Chapter 2, Baking Our Poky-based System, introduces the environment needed for the first build.

Chapter 3, Using Hob to Bake an Image, shows the user-friendly graphical interface that can be used as a wrapper for configuration and as a build tool.

Chapter 4, Grasping the BitBake Tool, presents the first concepts and premises of the tool used to control all other pieces of the Yocto Project.

Chapter 5, Detailing the Temporary Build Directory, details the output directory tree of a build with focus on the tmp directory.

Chapter 6, Assimilating Packaging Support, introduces the package concepts and details the packaging support used by the Yocto Project.

Chapter 7, Diving into BitBake Metadata, details the concepts and syntaxes used by the Yocto Project metadata, both in recipes and configuration files.

Chapter 8, Developing with the Yocto Project, details how to use the Yocto Project to generate a custom development environment.

Chapter 9, Debugging with the Yocto Project, details which debug tools the Yocto Project provides and how to use them.

For More Information:
Chapter 10, Exploring External Layers, explores one of the most important concepts of the Yocto Project, which is the flexibility of using external layers.

Chapter 11, Creating Custom Layers, practices the steps of creation of layers.

Chapter 12, Customizing Existing Recipes, lists the common use cases of recipe customization and how to achieve them properly.

Chapter 13, Achieving GPL Compliance, summarizes the tasks and concepts involved in a copyleft compliance product.

Chapter 14, Booting Our Custom Embedded Linux, uses a real hardware machine together with the Yocto Project's tools.

Appendix, References, lists the references used in the book.
Meeting the Yocto Project

In this chapter, we will be introduced to the Yocto Project. The main concepts of the project, which are constantly used throughout the book, are discussed here. We will discuss the Yocto Project history, OpenEmbedded, Poky, BitBake, and Metadata in brief, so fasten your seat belt and welcome aboard!

What is the Yocto Project?
The Yocto Project is a Linux Foundation workgroup defined as:

"The Yocto Project provides open source, high-quality infrastructure and tools to help developers create their own custom Linux distributions for any hardware architecture, across multiple market segments. The Yocto Project is intended to provide a helpful starting point for developers."

The Yocto Project is an open source collaboration project that provides templates, tools, and methods to help us create custom Linux-based systems for embedded products regardless of the hardware architecture. Being managed by a Linux Foundation fellow, the project remains independent of its member organizations that participate in various ways and provide resources to the project.

It was founded in 2010 as a collaboration of many hardware manufacturers, open source operating systems, vendors, and electronics companies in an effort to reduce their work duplication, providing resources and information catering to both new and experienced users.

Among these resources is OpenEmbedded-Core, the core system component, provided by the OpenEmbedded project.

For More Information:
Meeting the Yocto Project

The Yocto Project is, therefore, a community open source project that aggregates several companies, communities, projects, and tools, gathering people with the same purpose to build a Linux-based embedded product; all these components are in the same boat, being driven by its community needs to work together.

Delineating the Yocto Project

To ease our understanding of the duties and outcomes provided by the Yocto Project, we can use the analogy of a computing machine. The input is a set of data that describes what we want, that is, our specification. As an output, we have the desired Linux-based embedded product.

If the output is a product running a Linux-based operating system, the result generated is the pieces that compose the operating system, such as the Linux kernel, bootloader, and the root filesystem (rootfs) bundle, which are properly organized.

To produce the resultant rootfs bundle and other deliverables, the Yocto Project's tools are present in all intermediary steps. The reuse of previously built utilities and other software components are maximized while building other applications, libraries, and any other software components in the right order and with the desired configuration, including the fetching of the required source code from their respective repositories such as The Linux Kernel Archives (www.kernel.org), GitHub, and www.SourceForge.net.

Preparing its own build environment, utilities, and toolchain, the amount of host software dependency is reduced, but a more important implication is that the determinism is considerably increased. The utilities, versions, and configuration options are the same, minimizing the number of host utilities to rely on.

We can list some projects, such as Poky, BitBake, and OpenEmbedded-Core, under the Yocto Project umbrella, all of them being complimentary and playing specific roles in the system. We will understand exactly how they work together in this chapter and throughout the book.

Understanding Poky

Poky is the Yocto Project reference system and is composed of a collection of tools and metadata. It is platform-independent and performs cross-compiling, using the BitBake tool, OpenEmbedded Core, and a default set of metadata, as shown in the following figure. It provides the mechanism to build and combine thousands of distributed open source projects to form a fully customizable, complete, and coherent Linux software stack.

For More Information:
Poky's main objective is to provide all the features an embedded developer needs.

**Using BitBake**

**BitBake** is a task scheduler that parses Python and Shell Script mixed code. The code parsed generates and runs tasks, which are basically a set of steps ordered according to the code's dependencies.

It evaluates all available configuration files and recipe data (known as **metadata**), managing dynamic variable expansion, dependencies, and code generation. It keeps track of all tasks being processed in order to ensure completion, maximizing the use of processing resources to reduce build time and being predictable. The development of BitBake is centralized in the **bitbake-devel@lists.openembedded.org** mailing list, and its code can be found in the **bitbake** subdirectory of Poky.

**OpenEmbedded-Core**

The **OpenEmbedded-Core** metadata collection provides the engine of the Poky build tool. It is designed to provide the core features and needs to be as clean as possible. It provides support for five different processor architectures (**ARM**, **x86**, **x86-64**, **PowerPC**, **MIPS** and **MIPS64**), supporting only QEMU-emulated machines.

The development is centralized in the **openembedded-core@lists.openembedded.org** mailing list, and houses its metadata inside the **meta** subdirectory of Poky.
Metadata

The metadata, which is composed of a mix of Python and Shell Script text files, provides a tremendously flexible system. Poky uses this to extend OpenEmbedded-Core and includes two different layers, which are another metadata subset shown as follows:

- **meta-yocto**: This layer provides the default and supported distributions, visual branding, and metadata tracking information (maintainers, upstream status, and so on)
- **meta-yocto-bsp**: This layer, on top of it, provides the hardware reference boards support for use in Poky

*Chapter 7, Diving into BitBake Metadata*, explores the metadata in more detail and serves as a reference when we write our own recipes.

The alliance of OpenEmbedded Project and Yocto Project

The **OpenEmbedded** project was created around January 2003 when some core developers from the **OpenZaurus** project started to work with the new build system. The OpenEmbedded build system has been, since its beginning, a tasks scheduler inspired and based on the **Gentoo Portage** package system named BitBake. The project has grown its software collection, and a number of supported machines at a fast pace.

As consequence of uncoordinated development, it is difficult to use OpenEmbedded in products that demand a more stable and polished code base, which is why Poky was born. Poky started as a subset of OpenEmbedded and had a more polished and stable code base across a limited set of architectures. This reduced size allowed Poky to start to develop highlighting technologies, such as IDE plugins and QEMU integration, which are still being used today.

Around November 2010, the Yocto Project was announced by the Linux Foundation to continue this work under a Linux Foundation-sponsored project. The Yocto Project and OpenEmbedded Project consolidated their efforts on a core build system called OpenEmbedded-Core, using the best of both Poky and OpenEmbedded, emphasizing an increased use of additional components, metadata, and subsets.

For More Information:

Summary

This first chapter provided an overview on how the OpenEmbedded Project is related to the Yocto Project, the components which form Poky, and how it was created. In the next chapter, we will be introduced to the Poky workflow with steps to download, configure, and prepare the Poky build environment, and how to have the very first image built and running using QEMU.
Where to buy this book


Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.