

OPEN SOURCE SOFTWARE DEVELOPMENT SERIES

Embedded Linux Systems with the Yocto Project™



Rudolf J. Streif

Embedded Linux Systems with the Yocto Project[™]

Embedded Linux Systems with the Yocto Project

Table of Contents

Cover

Title Page

Copyright Page

Contents

Foreword

Preface

Acknowledgments

About the Author

1 Linux for Embedded Systems

1.1 Why Linux for Embedded Systems?

1.2 Embedded Linux Landscape

1.2.1 Embedded Linux Distributions

1.2.2 Embedded Linux Development Tools

1.3 A Custom Linux Distribution Why Is It Hard?

1.4 A Word about Open Source Licensing

1.5 Organizations, Relevant Bodies, and Standards

1.5.1 The Linux Foundation

1.5.2 The Apache Software Foundation

1.5.3 Eclipse Foundation

1.5.4 Linux Standard Base

1.5.5 Consumer Electronics Workgroup

Table of Contents

1.6 Summary

1.7 References

2 The Yocto Project

2.1 Jumpstarting Your First Yocto Project Build

2.1.1 Prerequisites

2.1.2 Obtaining the Yocto Project Tools

2.1.3 Setting Up the Build Host

2.1.4 Configuring a Build Environment

2.1.5 Launching the Build

2.1.6 Verifying the Build Results

2.1.7 Yocto Project Build Appliance

2.2 The Yocto Project Family

2.3 A Little Bit of History

2.3.1 OpenEmbedded

2.3.2 BitBake

2.3.3 Poky Linux

2.3.4 The Yocto Project

2.3.5 The OpenEmbedded and Yocto Project Relationship

2.4 Yocto Project Terms

2.5 Summary

2.6 References

3 OpenEmbedded Build System

3.1 Building Open Source Software Packages

3.1.1 Fetch

3.1.2 Extract

3.1.3 Patch

3.1.4 Configure

3.1.5 Build

Table of Contents

3.1.6 Install

3.1.7 Package

3.2 OpenEmbedded Workflow

3.2.1 Metadata Files

3.2.2 Workflow Process Steps

3.3 OpenEmbedded Build System Architecture

3.3.1 Build System Structure

3.3.2 Build Environment Structure

3.3.3 Metadata Layer Structure

3.4 Summary

3.5 References

4 BitBake Build Engine

4.1 Obtaining and Installing BitBake

4.1.1 Using a Release Snapshot

4.1.2 Cloning the BitBake Development Repository

4.1.3 Building and Installing BitBake

4.2 Running BitBake

4.2.1 BitBake Execution Environment

4.2.2 BitBake Command Line

4.3 BitBake Metadata

4.4 Metadata Syntax

4.4.1 Comments

4.4.2 Variables

4.4.3 Inclusion

4.4.4 Inheritance

4.4.5 Executable Metadata

4.4.6 Metadata Attributes

4.4.7 Metadata Name (Key) Expansion

Table of Contents

4.5 Source Download

4.5.1 Using the Fetch Class

4.5.2 Fetcher Implementations

4.5.3 Mirrors

4.6 HelloWorldBitBake Style

4.7 Dependency Handling

4.7.1 Provisioning

4.7.2 Declaring Dependencies

4.7.3 Multiple Providers

4.8 Version Selection

4.9 Variants

4.10 Default Metadata

4.10.1 Variables

4.10.2 Tasks

4.11 Summary

4.12 References

5 Troubleshooting

5.1 Logging

5.1.1 Log Files

5.1.2 Using Logging Statements

5.2 Task Execution

5.2.1 Executing Specific Tasks

5.2.2 Task Script Files

5.3 Analyzing Metadata

5.4 Development Shell

5.5 Dependency Graphs

5.6 Debugging Layers

5.7 Summary

Table of Contents

6 Linux System Architecture

6.1 Linux or GNU/Linux?

6.2 Anatomy of a Linux System

6.3 Bootloader

6.3.1 Role of the Bootloader

6.3.2 Linux Bootloaders

6.4 Kernel

6.4.1 Major Linux Kernel Subsystems

6.4.2 Linux Kernel Startup

6.5 User Space

6.6 Summary

6.7 References

7 Building a Custom Linux Distribution

7.1 Core ImagesLinux Distribution Blueprints

7.1.1 Extending a Core Image through Local Configuration

7.1.2 Testing Your Image with QEMU

7.1.3 Verifying and Comparing Images Using the Build History

7.1.4 Extending a Core Image with a Recipe

7.1.5 Image Features

7.1.6 Package Groups

7.2 Building Images from Scratch

7.3 Image Options

7.3.1 Languages and Locales

7.3.2 Package Management

7.3.3 Image Size

7.3.4 Root Filesystem Types

7.3.5 Users, Groups, and Passwords

7.3.6 Tweaking the Root Filesystem

Table of Contents

7.4 Distribution Configuration

7.4.1 Standard Distribution Policies

7.4.2 Poky Distribution Policy

7.4.3 Distribution Features

7.4.4 System Manager

7.4.5 Default Distribution Setup

7.5 External Layers

7.6 Hob

7.7 Summary

8 Software Package Recipes

8.1 Recipe Layout and Conventions

8.1.1 Recipe Filename

8.1.2 Recipe Layout

8.1.3 Formatting Guidelines

8.2 Writing a New Recipe

8.2.1 Establish the Recipe

8.2.2 Fetch the Source Code

8.2.3 Unpack the Source Code

8.2.4 Patch the Source Code

8.2.5 Add Licensing Information

8.2.6 Configure the Source Code

8.2.7 Compile

8.2.8 Install the Build Output

8.2.9 Setup System Services

8.2.10 Package the Build Output

8.2.11 Custom Installation Scripts

8.2.12 Variants

8.3 Recipe Examples

Table of Contents

- 8.3.1 C File Software Package
- 8.3.2 Makefile-Based Software Package
- 8.3.3 CMake-Based Software Package
- 8.3.4 GNU Autotools-Based Software Package
- 8.3.5 Externally Built Software Package

8.4 Devtool

- 8.4.1 Round-Trip Development Using Devtool
- 8.4.2 Workflow for Existing Recipes

8.5 Summary

8.6 References

9 Kernel Recipes

9.1 Kernel Configuration

- 9.1.1 Menu Configuration
- 9.1.2 Configuration Fragments

9.2 Kernel Patches

9.3 Kernel Recipes

- 9.3.1 Building from a Linux Kernel Tree
- 9.3.2 Building from Yocto Project Kernel Repositories

9.4 Out-of-Tree Modules

- 9.4.1 Developing a Kernel Module
- 9.4.2 Creating a Recipe for a Third-Party Module
- 9.4.3 Including the Module with the Root Filesystem
- 9.4.4 Module Autoloading

9.5 Device Tree

9.6 Summary

9.7 References

10 Board Support Packages

10.1 Yocto Project BSP Philosophy

Table of Contents

10.1.1 BSP Dependency Handling

10.2 Building with a BSP

10.2.1 Building for the BeagleBone

10.2.2 External Yocto Project BSP

10.3 Inside a Yocto Project BSP

10.3.1 License Files

10.3.2 Maintainers File

10.3.3 README File

10.3.4 README.sources File

10.3.5 Prebuilt Binaries

10.3.6 Layer Configuration File

10.3.7 Machine Configuration Files

10.3.8 Classes

10.3.9 Recipe Files

10.4 Creating a Yocto Project BSP

10.4.1 Yocto Project BSP Tools

10.4.2 Creating a BSP with the Yocto Project BSP Tools

10.5 Tuning

10.6 Creating Bootable Media Images

10.6.1 Creating an Image with Cooked Mode

10.6.2 Creating an Image with Raw Mode

10.6.3 Kickstart Files

10.6.4 Kickstart File Directives

10.6.5 Plugins

10.6.6 Transferring Images

10.7 Summary

10.8 References

11 Application Development

Table of Contents

11.1 Inside a Yocto Project ADT

11.2 Setting Up a Yocto Project ADT

11.2.1 Building a Toolchain Installer

11.2.2 Installing the Toolchain

11.2.3 Working with the Toolchain

11.2.4 On-Target Execution

11.2.5 Remote On-Target Debugging

11.3 Building Applications

11.3.1 Makefile-Based Applications

11.3.2 Autotools-Based Applications

11.4 Eclipse Integration

11.4.1 Installing the Eclipse IDE

11.4.2 Integrating a Yocto Project ADT

11.4.3 Developing Applications

11.4.4 Deploying, Running, and Testing on the Target

11.5 Application Development Using an Emulated Target

11.5.1 Preparing for Application Development with QEMU

11.5.2 Building an Application and Launching It in QEMU

11.6 Summary

11.7 References

12 Licensing and Compliance

12.1 Managing Licenses

12.1.1 License Tracking

12.1.2 Common Licenses

12.1.3 Commercially Licensed Packages

12.1.4 License Deployment

12.1.5 Blacklisting Licenses

12.1.6 Providing License Manifest and Texts

12.2 Managing Source Code

Table of Contents

12.3 Summary

12.4 References

13 Advanced Topics

13.1 Toaster

13.1.1 Toaster Operational Modes

13.1.2 Toaster Setup

13.1.3 Local Toaster Development

13.1.4 Toaster Configuration

13.1.5 Toaster Production Deployment

13.1.6 Toaster Web User Interface

13.2 Build History

13.2.1 Enabling Build History

13.2.2 Configuring Build History

13.2.3 Pushing Build History to a Git Repository Server

13.2.4 Understanding the Build History

13.3 Source Mirrors

13.3.1 Using Source Mirrors

13.3.2 Setting Up Source Mirrors

13.4 Autobuilder

13.4.1 Installing Autobuilder

13.4.2 Configuring Autobuilder

13.5 Summary

13.6 References

A: Open Source Licenses

A.1 MIT License (MIT)

A.2 GNU General Public License (GPL) Version 2

A.3 GNU General Public License (GPL) Version 3

A.4 Apache License Version 2.0

Table of Contents

B: Metadata Reference

Index