سرفصل آموزشی

پک پیشرفته مهندسی Embedded Linux

فهرست سرفصل‌های دوره‌های آموزشی

سرفصل‌های دوره آموزشی

Embedded Linux

Level-3

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

Introduction to bulidroot

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

Basic Buildroot usage

- Getting and setting up Buildroot
- Configuring and building a basic system with Buildroot for the Apex-V210
- Flash and test the generated system on the Apex-V210

Managing the build and configuration

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

Buildroot source and build trees

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

Toolchains in Builtroot

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

Managing the Linux kernel configuration

- Loading, changing and saving the kernel configuration

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

- Rootfile systems 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

Download infrastructure in Buildroot

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

GNU Make 101

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

Integrating 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
- New packages in Buildroot
- Create a new package for nInvaders
- Understand how to add dependencies
- Add patches to nInvaders for Nunchuk support

Advanced package aspects

- 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

Advanced 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

Analyzing the build: licensing, dependencies, build time

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

Advanced topics

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

Advanced aspects in action

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

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

Application development with Buildroot in action

- Build and run your own application
- Remote debug your application
- Use <pkg>_OVERRIDE_SRCDIR
- Set up Eclipse for Buildroot application development

Understanding Buildroot internals

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

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

سرفصل‌های دوره آموزشی

Yocto Project

& OpenEmbedded

Introduction to embedded Linux build systems

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

Overview of the Yocto Project and the Demo - First Yocto Project build

- Organization of the project source tree
- Building a root filesystem image using the Yocto Project
- Downloading the Poky reference build system
- Building a system image

Using Yocto Project – basics

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

Flashing and booting

- Flashing and booting the image on the board

Using Yocto Project - advanced usage Demo - Using NFS and configuring the build

- Configuring the build system
- Customizing the package selection
- Configuring the board to boot over NFS
- Learn how to use the PREFERRED_PROVIDER mechanism

Writing recipes - basics Demo - Adding an application to the build

- Writing a minimal recipe
- Adding dependencies
- Development workflow with bitbake
- Writing a recipe for nInvaders
- Adding nInvaders to the final image

Writing recipes - advanced features

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

Layers Demo - Writing a layer

- What layers are
- Where to find layers
- Creating a layer
- Learn how to write a layer
- Add the layer to the build
- Move nInvaders to the new layer

Writing a BSP Demo - Implementing the kernel changes

- Extending an existing BSP
- Adding a new machine Bootloaders
- Linux and the linux-yocto recipe
- Adding a custom image type
- Extend the kernel recipe to add the MPU6050 accelerometer/gyro driver
- Configure the kernel to compile the MPU6050 accelerometer/gyro driver
- Play nInvaders

Creating a custom image Demo - Creating a custom image

- Writing an image recipe
- Adding users/groups
- Adding custom configuration
- Writing and using package groups recipes
- Writing a custom image recipe
- Adding nInvaders to the custom image

Creating and using an SDK Demo - Experimenting with the SDK

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

Questions and Answers

- Questions and answers with the audience about the course topics
- Extra presentations if time is left, according what most participants are interested in

سرفصل‌های دوره آموزشی

C Programming

& App Development

Part one - Essential C programming

Introduction

C Fundamentals

Formatted I/O

Expression & Statement

Loops

Basic Types

Arrays

Functions

- Programming Process
- Setting Up
- Specification
- Code Design
- Prototype
- Makefile
- Testing
- Debugging
- Maintenance
- Revisions
- Electronic Archaeology
- Marking Up the Program
- Using the Debugger
- Text Editor as a Browser
- Add Comments

Pointers

Strings

The Preprocessor

Part Two – Advance C Programming

Writing Large Program

Advance uses of pointers

Declarations

Structure

Union

Enumeration

Program Design

Low-Level Programming

The Standard C library

- Input/output
- Numbers and Characters
- Error Handling
- C99

Part Three – Linux Programming

Memory Management and Allocation

File I/O

Time

Processes

Pipes and Fifo’s

Signals

POSIX Threads

Writing Secure Privileged Program

Inter Process Communication

Mutex

Networking and Sockets

Sockets – Fundamentals of TCP/IP networks

Part Four – Secure C Programming

Introduction

Secure working with strings

Secure pointers

Dynamic memory management

Integer Security

Formatted Output

Concurrency

File I/O

Recommended Practice

- The security development life-cycle
- Design
- Implementation
- Verification