Embedded Linux Development with Yocto

Course Syllabus

Course Title

Embedded Linux Development with Yocto

Course Overview

Linux OS has been one of the defacto Operating Systems deployed in a gamut of embedded devices, but the Linux development ecosystem has been fragmented and discreet. This brings about a plethora of challenges in terms of long-term support, maintenance, Licensing issues, incompatible distribution, and development systems to name a few. To overcome these challenges, Linux Foundation has spearheaded the Yocto Project with an objective to simplify the software development process for Linux distributions. The Yocto Framework is an open-source collaboration with a huge community support specially designed to customize Linux image for deployment in Embedded and IoT application.

This course introduces you to the Yocto Project and its deployment on hardware boards for embedded Linux development. If you are an embedded Linux enthusiast and would like to learn and use the various features offered by the Yocto Project, then this is the course for you. At the end of this course, you would have gained mastery over the Yocto Linux framework and create custom distributions for COTS boards and seamlessly deploy them on real-time platforms

Course Duration

40+ hrs – 17hr Lecture, 23hr Lab Sessions

Course Eligibility

- Aspiring graduates seeking job opportunities in embedded Linux development
- Professionals interested in up-skilling themselves in embedded Linux domain
- Embedded Linux developers who wish to become experts in using Yocto

Course Highlights

- Course delivered by seasoned industry experts with more than 17 years of experience in embedded software design and development
- Course contents designed exclusively to give deep insights into the Yocto framework and enable participants to create custom Yocto distributions for custom boards
- Well-structured and modular program framework to ensure participants gain mastery in embedded system design and development concepts
- Online and interactive sessions with a balanced mix of theory, hands-on assignments and mini Projects

Course Pre-requisites

- Ubuntu or Linux host PC for development
- Basic Knowledge of Linux commands – command memento and vi memento
- Basic programming skills in C and Assembly Language
- Good Know-How of working with embedded COTS boards

Course Requirements

- BeagleBone Black
- Ubuntu 20.04 LTS Host Machine
- Candidates must have a laptop/desktop with minimum configuration: - 64-bit processor - 8GB RAM - 200GB HDD space - Minimum 2Mbps internet connection

Course Contents

1. Overview of Embedded Linux Systems
2. Introduction to Yocto Project
3. Embedded Linux Development
4. Yocto Build System Concepts and Terminologies
5. Customization of Yocto BSP for BeagleBone Black

Module Contents

Module - 1

- **Module Title**
  Overview of Embedded Linux Systems

- **Module Overview**
  This module begins with a brief introduction to Embedded Linux Systems and their applications. We will understand the different components of a Linux systems and the tasks performed by each of these components during boot and runtime process. Next, we shall explore the various Linux distributions available for development and deployment onto Embedded platforms. We shall investigate the limitations and challenges each of these distributions pose and how the Yocto project overcomes these challenges by providing a comprehensive solution

- **Module Topics**

- Introduction to Embedded Linux Systems
- Components of an Embedded Linux System – BootROM, FSBL, SBL, Kernel, RootFS and Application Binaries
- Overview of Embedded Linux Distributions
- Challenges and Limitations of Embedded Linux Build Systems
- Yocto Project as a solution to these challenges

Module - 2

- **Module Title**
  Introduction to Yocto Project

- **Module Overview**
  This module begins with introduction to the Yocto project with a brief overview on its history, contributors, maintainers, adopters and release lifecycles. We next look into the steps for setting up the host machine build environment for building the Linux image for the BeagleBone Black board from its Yocto sources. We shall deploy the built image onto the target hardware board and explore the Linux boot sequence in detail. We shall also dwell deep into the Yocto source code framework and the significance of the different directories downloaded and created during the image build process

- **Module Topics**
  - Introduction to Yocto project and its release process and lifecycle
  - Setup of Yocto Host Build environment
  - Yocto source for BB Black
    - Machine and Distro Type
    - Yocto Image Types
    - Building and Flashing Yocto image to target board
    - Running Yocto image on target board
    - Understanding the Linux Boot process in BB Black
  - Understanding the Yocto source code framework
    - Exploring Build and source directories

Module - 3

- **Module Title**
  Embedded Linux Development

- **Module Overview**
  This module aims at providing the necessary insights into the Uboot and Kernel source code framework for the BB black Yocto repository. We shall analyse the most important configuration and

device tree files in these sources and understand its relevance to the hardware counterpart. We shall also have a hands-on with the basic Linux commands for verifying the system and interfaces on board once its booted up to the Linux shell. This module shall also enable the participants to verify the various subsystems like memory, connectivity, display, low-speed peripherals, and storage at the kernel level

- **Module Topics**
  - Embedded Linux Boot Sequence
  - Overview of Uboot Bootloader - FSBL, SBL
    - Uboot source code structure - defconfig, DTS, drivers, menuconfig
    - Bootargs, Environment Variables, Uboot commands
    - Uboot Power-On-Self-Test (POST)
    - Boot from SD, MMC
    - Boot from TFTP, NFS
  - Overview of Linux Kernel
    - Linux kernel source code structure - config, DTS, menuconfig, drivers
    - DTS basics - pinmux, clocks, interrupt, memory
    - Building kernel image and kernel configuration customization
    - Loadable kernel modules
    - Booting kernel
    - Kernel commands - insmod, lsmod, dmesg
    - Basics of Linux device drivers
  - Overview of Root file systems
  - Verifying Interfaces at Kernel
    - dmesg, procfs, sysfs, debugfs
    - Verifying USB, SD, UART, GPIO
    - Verfying Display, Camera
    - Verifying Security, GPU, VPU
    - Verifying PCIe, Ethernet
    - Verifying I2C, I2S

Module - 4

- **Module Title**
  Yocto Build System Concepts and Terminologies

- **Module Overview**
  This module provides an in-depth understanding and hands-on on the Yocto Build system concepts and terminologies. We shall cover the details of meta-layers and recipes in Yocto sources and learn how to build and modify a recipe using the bitbake and devtools. We shall also look into the various log files created during the build process which aids in checking for results or errors during the build

process. We shall also cover the basics of the Files and packages and learn how to declared runtime and build time dependencies for the various packages that are included with the image. Finally, we shall learn how to declare the preferred providers for Uboot and Kernel sources in the Yocto framework

- **Module Topics**
  - Understanding Meta Layers, Recipes, Classes, Configuration files
  - Using Bitbake and Devtool Tool
  - Overview of Variables used in Recipes (WORKDIR, S, D, PN, PV, PR)
  - Exploring runtime log files
  - Understanding FILES and PACKAGES
  - Overview of Build and Runtime Dependencies - DEPENDS and RDEPENDS
  - RPROVIDES , virtual targets and PREFERRED_PROVIDER

Module - 5

- **Module Title**
  Customization Yocto BSP for BeagleBone Black

- **Module Overview**
  After having learnt the intricacies of the Yocto build system and the interfaces of Embedded Linux environment on target board, it is time to start customizing and creating our own distributions for existing or new hardware platforms. The aim of this module is to provide the required knowledge and hands-on to achieve the same. We begin with learning how to create a new meta-layer and add our own recipes to the meta-layer. We shall also learn on appending new features to existing meta-layers and recipes thus enabling us to create a custom distribution for a specific hardware board. We shall learn more about package group and package management in Yocto. Finally, we shall learn the procedure to modify, enhance and customize the Uboot and kernel sources of BB Black and create and integrate these patches with the Yocto build system.

- **Module Topics**
  - Creating a custom meta-layer
  - Creating a custom recipe and Appending recipe
  - Creating a custom distribution
  - Adding packages to the image
  - Creating a package group and package management
  - Creating a patch for Uboot and Linux Kernel