DOWNLOAD PDF INTRODUCTION TO EMBEDDED LINUX

Chapter 1 : Introduction to Embedded Linux - Online Course - IEEE Boston

Introduction to Embedded Linux The use of Linux in embedded products has skyrocketed in the last decade. In this free webinar, you will learn the basics of getting started in Embedded Linux Development and a short overview of best practices.

The basic architecture of embedded Linux system The core part of any operating system is the kernel. The kernel is responsible for the particular features which operating systems possess. It makes the standard operating system entirely different from the infinite loop operating system. Architecture of a standard Linux Kernel Kernel Subsystems: WhyARM hardware is widely used? The term OS porting means to modify or customize an OS, which is running on particular hardware architecture, in such a way that it can run on another particular kind of architecture when loaded into one. We can find Linux OS normally in personal computers. The Linux supporting personal computer mostly use Intel processors, and the architecture of Intel based processors are x86 or x But as we mentioned before, most of the embedded devices use ARM based processors. So if we required loading the Linux OS into an embedded device, we must do something that will enable the OS running on x86 based hardware to run on ARM based hardware, and we call it Linux Porting. Simply, loading and running the Linux from one processor to another processor of different architecture type. After the porting the general purpose Linux changes to a customized task specific OS. Linux porting is a wide topic itself, and is the most important step in developing an embedded Linux system. It is also the most difficult step as well. I will try to explain the basics of Linux porting briefly. Necessary things to be taken care of while Linux porting 3. Linux support wide variety of hardware architectures, including ARM architecture. It is the first code that execute when the device is powered up. Normally the firmware is just a simple initialization and bootloader routine. It is considered as deeply embedded. It is the first code that should be ported to a new platform. The choice of firmware depends upon the complexity of the operating system. The main function of the firmware is to load and boot an operating system. Firmware can remain active even if the system is running. It supportvarious ARM boards and processors. Hence the firmware helps to make the porting a lot more easier. The operation of the firmware can be described through the following steps Boot Loader 3. U-Boot is the most popular and actievely developed bootloader. Usually a compiler runs in a machine of specific architecture compiles code for the same architecture itself. A compiler does the following steps A cross is a kind of compiler program which runs in a system of particular hardware architecture, but can compile codes and generate executable for systems having different hardware architecture. The compiling using a cross compiler is called cross compiling of the source code. GCC based cross compiler tool chains are widely used and are available for free.

Chapter 2 : Chapter 3: Exploring Embedded Linux Systems â€“ Exploring Raspberry Pi

Agenda Introduction: open-source and free software principles, advantages in the embedded space, hardware needed for embedded Linux Open Source for embedded systems: tools, bootloaders, kernel, system.

We provide you with three 3 speed choices that you can select when watching a video. If the video is buffering then the first thing you want to click on a slower speed option found at the bottom of the video window. Try clicking on Medium and Low and see if that allows the video to play without buffering. If it does not, please read through the rest of this FAQ and follow the steps outlined below. Video will not play: In order to play our videos, you have to have Adobe Flash Player installed. If you click on the video and nothing happens, then you do not have Adobe Flash Player installed. Click on this link to download and install it. If you have Adobe Flash Player installed and your video is not playing, complete the following steps: Go to your course page, select a video, and click play. Once the video begins playing, place your mouse over the video and right click. Click here , where you will see your version and what is the latest version. You need to have the latest version installed. If you do not have Adobe Flash Player, click here to download and install it. Please check to make sure that your pop-up blocker is turned off when you go to our website. This will prevent the video window from opening. Internet speed needs to be 2. You will want to check and see if your connection speed is 2. You can do this by clicking on this link and running a speed test. Jitter and Packet Loss: Your Jitter rate lets us know if your connection speed is consistent while your Packet Loss rate lets us know if your internet connection could be losing pieces of the video. This is measured by a letter scale with A being the best. If your Packet Loss is below a B, your connection is losing pieces of the video and causing the video to skip. If your Jitter is below a B, your connection speed is very inconsistent and may be changing while you are watching the video which will result in buffering issues. If either of these is the case you will need to contact your internet provider. You will need to have ports , and 80 open. You can run a port test at this link. If you are watching from work and those ports are not open, check with your IT department. If you are on a shared internet connection, test playing videos during non-peak hours when usage by others on your network is low. If you are using Chrome, use the instructions in this link to reset your browser. If you are using Internet Explorer IE you may need to reset it. You can do this by clicking on the cog icon on the top right of your page. Then click on tools, internet options, advanced and reset. If you are using Firefox, use the instructions in this link to reset your browser. Other things to check: At home or at work, test using different browsers and computers to see if you get the same result. Make sure you have the latest version of the browser you are using. Try from a different location using a different network. If you are on a wireless connection try a wired solution. If you are on a home network: Test playing the videos when you are the only user on the network. Try by-passing your router to eliminate it as the point of failure. A direct plug from your ISP to your computer will take care of this and will make sure that neither your router nor other users are affecting your connection. If you are running Firefox, you can check your plugins here: File sharing and peer to peer sharing programs, even when limited to low bandwidth, can make it almost impossible to stream properly. Tracking Progress on Mobile Devices: The GogoTraining site is designed to use the native player found in your browser. If you are taking your course on a mobile device and want to make sure your progress is tracked, then you will need to play the videos from the native player that is in your browser. Do not download any third party apps to play the videos as they do not allow your progress to be recorded. If you have installed a third party app and want to receive a Course Completion Certificate, then you will need to uninstall the 3rd party app or see if the app has the ability for you to disable it for certain sites. Submitting Debugging Information If you have tried everything else and still are having trouble viewing the videos, you can visit this page where you can help us debug your particular setup by following the instructions on the page and submit the debugging information to us.

Chapter 3 : Introduction to Embedded Linux

Linux, available for many architectures, is an obvious candidate for an embedded system, and it already is being used widely in this area. Its open nature makes it particularly attractive to developers.

Introduction to Embedded Linux Systems Today I begin to share the voyage that was my Master Thesis, in total it will be eight to ten blog posts, explaining the embedded Linux World and two of its distributions, Android and Tizen. This series is meant for embedded enthusiasts that wish to learn the basics to understand how embedded Linux and its distributions work. It also is meant to help anyone understand what have those raspberry pi images that we download from the web and how they work. Embedded Operating Systems in general can be divided into two categories: These last work at the lowest level of operating systems and are reserved to low end applications on low end devices such as medium range arduinos or arm boards. Finally there are the low level embedded systems that run without any formal operating system. They are designed from scratch for each purpose, like sensor controlling or mini automatons. An operating system is a piece of software that manages all hardware and software from a device. The advanced operating systems run on two different modes, user mode and kernel mode, the user mode is reserved to user and other low privilege applications. The kernel mode is where all the important components run such as device drivers, process management, etc. Considering all the applications that run in user mode as a single application, one can reduce the modern device architecture into three layers, as the above figure shows; a platform component that aggregates all the user data, applications and user interface; a kernel that controls all the hardware and manages all the system; and the hardware where all the software runs and all data is stored. The Linux based operating systems or distributions are merely a Linux kernel with an application component over it, like Android, Tizen or Ubuntu. Linux Kernel Components Linux kernel is composed by a variety of different components, some of the most important ones are shown in the following image. To an application, the kernel can be seen as a service provider, because the kernel knows how to read or write from the disk, how to create processes and threads or if a device was connected. Threads, processes and synchronization are handled by the kernel Process Man- agement component; this also handles scheduling and partition of tasks through all of the available Central Processing Unit CPU. What Linux has is a common set of operations that were implemented for a variety of known file systems types. The Device Driver component is where all the specific hardware interaction is made. The Board Support Package consists on the specific software needed to run the kernel on a single device. This component can be divided by architecture, micro controller family and board. Folder Structure The above image, represents the source root folder from a regular Linux kernel. On it, it is possible to find the location of the previous enumerated components. The Device Driver component is on the drivers folder; the VFS can be found in the fs; the Process Management on the kernel; Network Stack on the net folder; and finally the SCI can be found also on the kernel folder, but its architecture dependent source is inside the arch folder. The Board Support Package component can be found under the arch folder, comprising all non generic source code as well as the device configurations. The following image expands partially the arch folder. The direct arch sub-folders are architecture folders: VFS or Memory Management. The config folder is present in all architecture folders and contains the configurations for each supported machine. Building Linux kernel uses a series of hierarquial makefiles to compile its sources. These makefiles use a known set of environment variables to allow external configuration, but only two are mandatory: ARCH - The target architecture from the device which the kernel is going to be build for. The ARCH purpose is to select which of the architectures it should search for a config file and all architectures have a folder on its root called config containing all the configurations to the supported devices. After the environment initialization, shown in the above listing , the next step is preparing the build to the target device. This option copies the device configuration file from its respective folder into the kernel root and does some pre-build configuration. After all the setup the final step is to simply build the kernel image: Modules Linux kernel modules are pieces of software that the kernel can load or unload on demand depending on the current needs of the system. This modules work as a way to expand the kernel functionality without the need to recompile it, so they are standalone libraries and not part of the kernel image itself. Each new device can add different modules to the kernel e. After the modules are built, another command is executed to retrieve the deliverables: These modules will be later installed on the target device. Device Tree Blob Linux kernel has three ways to detect the device hardware. The first consists of having hardcoded on its sources every single hardware description that a target device contains. The second is having some kind of external service e. BIOS that auto detects the hardware and communicate them with the kernel. The DTB is the hardware description of a device, it consists on a binary file that is passed to the kernel in the boot phase. This allow to generate the kernel for multiple devices of the same family , where the selection is made depending on the DTB file passed by the bootloader. For instance, consider Wandboard and Sabre board, both have an i. MX6 pro- cessor, but their hardware is significantly different. At the boot phase the bootloader passes a wandboard or sabre DTB to the kernel, only then it knows the hardware that it will run upon. Wrap up It was addressed the different steps to build a Linux kernel from its sources, that should generate the following items: Platform The platform is the last component of an operating system. It consists of a collection of software designed to give the end users functionalities. The platform component is materialized as a set of images that can be deployed into a device, in one of which must exist a Root FileSystem rootfs. The rootfs contains all the files needed to make the system work correctly, it also handles the specific system initialization. Deployment This chapter addresses the different components that a Linux operating system has and which deliverables are associated to them: Linux can run from a wide array of options, like hard drive, flash memory, SD Card, etc. Regardless of the physical memory used, the device should have its storage divided in at least two partitions. The above image, illustrates a possible configuration of the non volatile memory containing a Linux operating system. The two mandatory partitions are the boot and rootfs. The boot partition, contains the bootloader, the dtb and the kernel images. The rootfs contains the main platform image and the kernel modules. Finally, the rest of the memory can be other partitions, for instance, if the platform deliverables have more than one image or just free space to be used by the system to store data.

Chapter 4 : Introduction to Embedded Linux (Hands-on training) Training Course

Course - Introduction to Embedded Linux Lecturer - Mike McCullough is President and CEO of RTETC, LLC. Mike has a BS in Computer Engineering and an MS in Systems Engineering from Boston University.

Chapter 5 : Introduction to Embedded Linux Three-Day Workshop (AMx) - Texas Instruments Wiki

Introduction to Embedded Linux in Theory and Practice - a Crash Course:The aim of this crash course is to provide a basic overview of embedded GNU/Linux.

Chapter 6 : Introduction to Debugging Embedded Linux Systems Training Series | TI Training

This course is an introduction to the Embedded Linux topics. It has no pre-requisites, so anybody can watch it, and it should hand hold you to understand the main buzz words around Linux.

Chapter 7 : ImaginationOverflow - Introduction to Embedded Linux Systems

An Introduction to Embedded Linux Development, Part 1 The traditional small, narrowly focused embedded systems retain their significant presence, but these newer arrivals can capitalize on embedding a full-featured operating system.

Chapter 8 : Embedded Linux Tutorial I Basics of Embedded Linux

Introduction to Embedded Linux - Module Booting Linux 01 - 1 Module Booting Linux Introduction This module begins by showing how a Linux distribution may be booted using a provided u-boot.

Chapter 9 : Embedded Linux Training - Embedded Linux Course Online | GogoTraining

The "Introduction to Embedded Linux" workshop was developed for engineers with embedded C/C++ programming experience who would like an overview of the Linux operating system from a practical standpoint centered around the development of embedded Linux applications and systems.