| **CL-2006 Operating System** | **LAB – 12**  **Embedded Linux** | |
| --- | --- | --- |
| **NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES**  **Spring 2023**  ***Objective:***  *In this lab, we’ll be discussing the following contents*   1. *What is an Embedded system?* 2. *Options for Embedded development?* 3. *Why embedded Linux?* 4. *What is QEMU and how does it help with embedded Linux?* 5. What is Raspberry Pi? 6. How can you use Raspberry Pi and QEMU to emulate embedded Linux?   ***Introduction:***  Some terminologies we will study in this lab:  ***Embedded Systems:***  Embedded systems are computer systems designed to perform specific functions within a larger system or product. They are typically integrated into devices or products that require a high degree of functionality and reliability, such as automotive systems, medical equipment, consumer electronics, and industrial control systems.  Unlike general-purpose computers, embedded systems are designed with specific hardware and software components that are optimized for a particular application. They are often built with microcontrollers or microprocessors, and may include sensors, actuators, and other hardware components that enable them to interact with the physical world.  ***Different Options for Embedded Development:***  When it comes to designing embedded systems, there are a number of systems that can be used each having its own limitations and trade-offs. Following are some of the more popular ones currently and historically.   1. **Real Time Operating Systems (RTOS)**    * Scheduler overhead    * More powerful microcontroller    * High hardware control    * Multi-threading, some common libraries    * Multi-tasking: networking, user interface etc. 2. **Embedded General Purpose Operating System (GPOS)**    * Large overhead (scheduler, memory management, background services).    * Microprocessor usually required (external memory such as RAM)    * Low hardware control.    * Multi-threading/processing, more libraries (community supported).    * Multi-tasking: networking, filesystem, GUI etc.    * Access to high-level languages.   ***Embedded Linux:***  Embedded Linux is a type of **GPOS** and refers to the use of the Linux operating system in embedded systems. Linux is a popular open-source operating system that is widely used in servers and desktop computers. However, due to its flexibility, reliability, and customization capabilities, Linux has become increasingly popular for use in embedded systems as well.  Embedded Linux is typically customized for specific hardware platforms and applications, with only the necessary software components included to reduce the overall system size and improve performance. The Linux kernel, which is the core of the operating system, can be configured to run on a wide range of hardware platforms, including microcontrollers, microprocessors, and system-on-chip (SoC) devices.  Some of the use cases include:   * Routers & light-weight servers. * Robotic Application. * Rockets & Spacecrafts. * Consumer electronics & Kiosks (ATM, Digital pumps etc).   **Why use embedded Linux?**  When working with the above use cases, you require an operating system to maintain the multi-tasking and control of hardware hence making a **GPOS** the best option as compared to other embedded systems.  Embedded Linux offers several advantages over other embedded operating systems, including its open-source nature, a wide range of available software libraries and tools, and a large developer community that provides support and contributes to the development of the operating system.  According to study in 2019, Linux was the most popular OS among embedded device developers and is still leading the charts today, and is in popular demand as the world moves towards automation.    ***QEMU:***  QEMU (short for "Quick EMUlator") is an open-source virtual machine emulator and virtualizer that can simulate a variety of hardware architectures and platforms, including x86, ARM, and MIPS. QEMU allows you to run a complete operating system and its applications inside a virtual machine, which can be very useful for testing, development, and debugging.  In the context of embedded Linux, QEMU can be used to emulate the target hardware platform on a host machine, allowing you to test and debug your embedded Linux system without having access to the physical hardware. This can be particularly useful during the early stages of development when hardware may not be available or is difficult to access.  QEMU can also be used to simulate different hardware configurations and architectures, allowing you to test your embedded Linux system on a range of devices without needing to have each device physically available. For example, you can use QEMU to test your embedded Linux system on an ARM-based device even if you only have access to x86-based hardware.  **Raspberry Pi:**  Raspberry Pi is a series of single-board computers developed by the Raspberry Pi Foundation, a UK-based charity organization. The Raspberry Pi computers are designed to be low-cost, credit-card-sized computers that can be used for a variety of applications, from education to industrial automation and from media centers to home automation systems.  The Raspberry Pi is based on an ARM processor and runs on Linux-based operating systems such as Raspbian, Ubuntu, and others. It features several input/output interfaces, including HDMI, USB, Ethernet, Wi-Fi, Bluetooth, and GPIO pins, allowing it to be connected to a wide range of devices and peripherals.  One of the key features of the Raspberry Pi is its low cost, which makes it accessible to a wide range of users, including students, hobbyists, and professionals. The Raspberry Pi has a large and active community of users and developers, who create and share software and hardware projects, tutorials, and resources.  The Raspberry Pi has been used for various applications, including educational projects, home automation systems, media centers, robotics, and IoT devices. Its versatility, low cost, and community support have made it a popular platform for learning and experimentation in the field of embedded systems and computer science.  **How can QEMU & Raspberry Pi help with Embedded Linux**  Here are some ways in which QEMU can help with embedded Linux on Raspberry Pi:  **Emulation of Raspberry Pi hardware:** QEMU can emulate the Raspberry Pi hardware, allowing you to test and debug your embedded Linux system on a virtual Raspberry Pi. This is useful when the actual hardware is not available or when you want to test your system on different hardware configurations.  **Cross-compilation:** QEMU can be used for cross-compiling code for the Raspberry Pi. You can develop and test your code on your development machine and then run it on the Raspberry Pi using QEMU.  **Debugging:** QEMU can help with debugging by allowing you to step through the code and identify issues in a simulated environment. This can be particularly useful when dealing with issues that only occur under specific conditions.  **Customization:** QEMU can be customized to emulate different Raspberry Pi models, including the Raspberry Pi Zero, Raspberry Pi 2, Raspberry Pi 3, and Raspberry Pi 4. This allows you to test your embedded Linux system on different hardware configurations and models.  Overall, QEMU is a powerful tool that can simplify the process of developing and testing embedded Linux systems for Raspberry Pi. By allowing you to test and debug your system in a virtual environment, QEMU can help you identify and fix issues early in the development process, saving time and reducing costs.  **Emulating RPI(Raspbian) on Ubuntu**  In-order to emulate RPI (Raspberry Pi)/Raspbian on your Ubuntu(Linux), follow given instructions step-by-step  1 – Before we start, you’re going to need 2 things   * Raspbian image that you can get from (get the Raspbian-jessie.zip) <http://downloads.raspberrypi.org/raspbian/images/raspbian-2017-04-10/> * QEMU kernel that you can get from([kernel-qemu-4.4.34-jessie](https://github.com/dhruvvyas90/qemu-rpi-kernel/blob/master/kernel-qemu-4.4.34-jessie))   Once you’re redirected to the link, click on **view raw** to download the kerel file   * Inside your Ubuntu VM, create a new folder:   ***mkdir ~/qemu\_vms/***   * Download and place the Raspbian Jessie image to **~/qemu\_vms/**. * Download and place the qemu-kernel to **~/qemu\_vms/** * Run ***sudo apt-get install qemu-system*** to install qemu-system * Unzip the jessie-image.zip file * Run ***fdisk -l <image-file>*** to see details of the img file (results would be sth like) | |  |
| * You see that the filesystem (.img2) starts at sector **92160(can be different for you)**. Now take that value and multiply it by 512, in this case it’s **512 \* 92160 = 47185920 bytes.** Use this value as an offset in the following command: * ***sudo mkdir /mnt/Raspbian*** * ***sudo mount -v -o offset=*47185920 *-t ext4 ~/qemu\_vms/<your-img-file.img> /mnt/Raspbian*** * You should see the following prompt      * Now, run the following command & comment all of the text within it and save ***sudo nano /mnt/raspbian/etc/ld.so.preload*** * Next, run the following command   ***sudo nano /mnt/raspbian/etc/fstab***   * **IF** you see anything with mmcblk0 in fstab, then:   + Replace the first entry containing /dev/mmcblk0p1 with /dev/sda1   + Replace the second entry containing /dev/mmcblk0p2 with /dev/sda2, save and exit.   **ELSE**   * + Do nothing * Now, run the following commands   + ***cd ~***   + ***sudo umount /mnt/Raspbian*** * At the end, run the command below to run **RPI** on **Ubuntu**   ***qemu-system-arm -kernel ~/qemu\_vms/<your-kernel-qemu> -cpu arm1176 -m 256 -M versatilepb -serial stdio -append "root=/dev/sda2 rootfstype=ext4 rw" -hda ~/qemu\_vms/<your-jessie-image.img> -net nic -net user,hostfwd=tcp::5022-:22 -no-reboot***   * Your RPI will be emulated directly on your Ubuntu without the need for actual hardware (as shown below) | |  |