Creating Your Own Operating System

C.Cara Evangeline
Computer Science and Engineering
National Institute of Technology
Email:caraevangeline10@gmail.com

C.Kushala
Computer Science and Engineering
National Institute of Technology
Email:kushala@gmail.com

I. Introduction

Writing an operating system is the most complicated task in the world of programming.We will write our own bootloader using 16-bit assembly language to create our own operating system. The first part of operating system is the Bootloader. Bootloader is a piece of program that runs before any operating system is running. It is used to boot other operating systems, usually each operating system has a set of bootloaders specific for it. Bootloaders usually contain several ways to boot the OS kernel and also contain commands for debugging and/or modifying the kernel environment. The bootloaders are generally written in 16-bit assembly(also called Real mode), then the bits can be extended to 32-bit(Protected mode). So the bootloaders must be written in 16-bit assembly.

II. Project Details

A. Requirements

You need an assembler that can convert your assembly instructions into raw binary format and an simulator to view. We are using Nasm assembler and Qemu simulator.

B. Procedure

We will create 4 stage OS. First is to just display message on the screen with colors, second is to take input from user,third and fourth for drawing graphics.

- Install Nasm and Qemu using the following commands:
  sudo apt–get install nasm
  sudo apt–get install qemu qemusystemx86_64
- 16–bit assembly language code is written in 4stageos.asm
- Type following command to compile file
  nasm –f bin 4stageos.asm –o 4stageos.bin
- Once file is compiled successfully and 4stageos.bin file is created, then run it in qemu.
  qemu–system–x86_64 4stageos.bin
- Load your os into the USB device using the following commands:
  lsblk
  sudo mkfs.vfat –I dev/sdb
  sudo dd if=4stageos.bin of=/dev/sdb
- The os stored in the USB device is booted into the virtualbox.
  1.First, connect the USB drive containing the operating system you want to boot to your virtualbox. Press Windows Key + X,and press Enter to open the Disk Management window.Look for the USB drive in the Disk Management window and note its disk number.
  2. open a Command Prompt as Administrator.Type the following command into the Command Prompt window and press Enter. This command will change to VirtualBoxs default installation directory. If you installed VirtualBox to a custom directory, youll need to replace the directory path in the command with the path to your own VirtualBox directory:
  cd %programfiles%\Oracle\VirtualBox
  3.Type the following command into the Command Prompt window, replacing # with the number of the disk you found above, and press Enter.
  VBoxManage internalcommands createrawvmdk –filename C:\usb.vmdk –rawdisk \\.\PhysicalDrive#
  4.Next, open VirtualBox as Administrator.
  5.Create a new virtual machine by clicking the New button and go through the wizard. Select the operating system on the USB drive when prompted.
  6.When you are asked to select a hard disk, select Use an existing virtual hard drive file, click the button to browse for the file, and navigate to it .
  7.Boot the virtual machine and it should boot the operating system from your USB drive, just as if you were booting it on normal computer.

III. Conclusion

Operating systems allow people to interact with computer hardware; they are made out of hundreds of thousands of lines of code. They are usually made with the C#, C, C++, and assembly programming languages. Operating systems allow you to navigate through a computer while creating storage and executing commands.
With the help of assembly language(nasm),a simple operating system is created and booted into the virtualbox using the USB device.
While it is possible to create an operating system in a language such as Pascal or BASIC, you will be better off using C or Assembly. Assembly is absolutely necessary, as some vital parts of an operating system require it. C++, on the other hand, contains keywords that need another fully-built OS to run.So here we have used Assembly language.
We have developed small things such as displaying text and interrupts .In extension of this project we will move on to things such as memory management and multitasking.