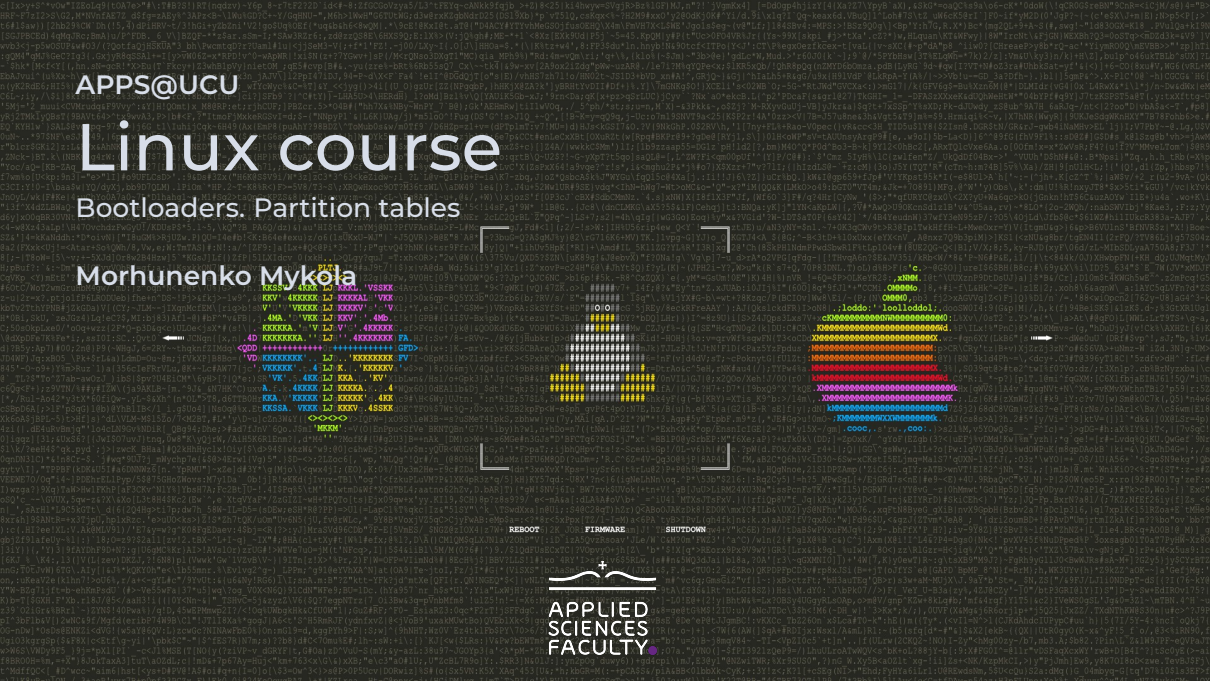
APPS@UCU

Linux course

Bootloaders. Partition tables

Morhunenko Mykola

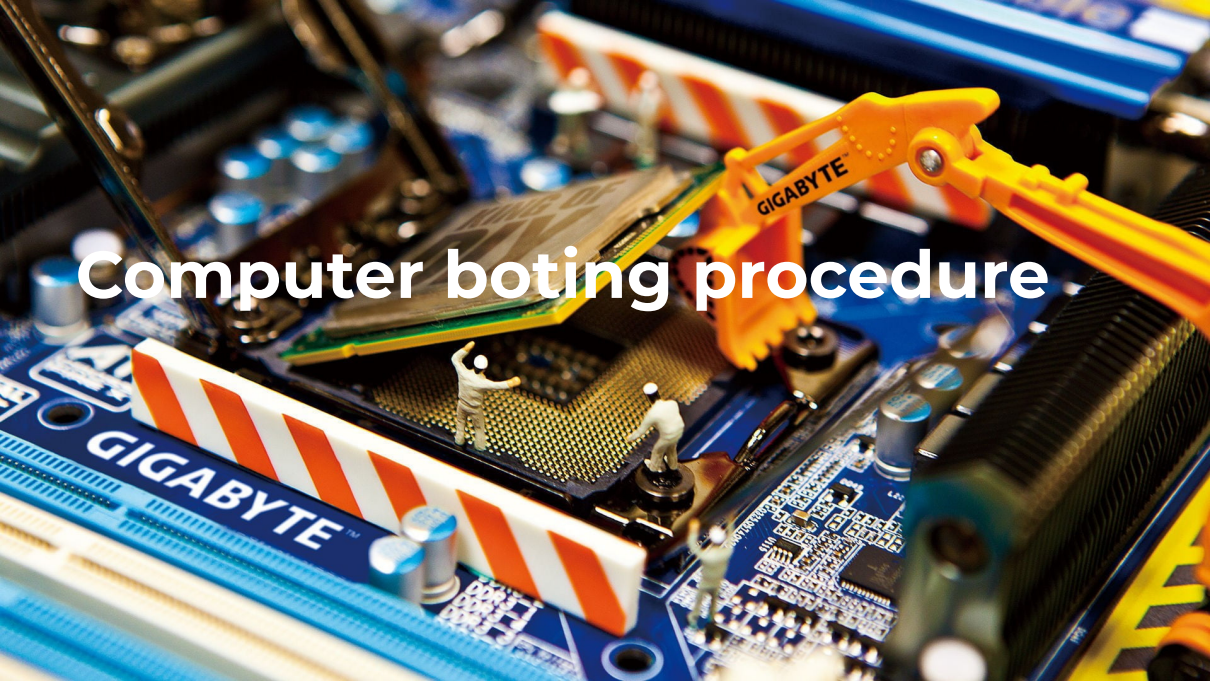


Introduction

- Next two topics are interconnected, and they also are important for understading how to manage the operating system
- Important terms:

  CMOS - Complementary Metal Oxide Semiconductor . Chip stores the settings like date and time, fan speed, booting sequence
  BIOS - Basic Input/Output System . Firmware to boot the computer
  UEFI - Unified Extensible Firmware Interface . Bootloader
  GRUB - GRand Unified Bootloader . Bootloader
  ESP - EFI System Partition
  GUID - Globally Unique Identifier (or UUID)
  MBR - Master Boot Record . Partition table
  GPT - GUID Partition Table . Partition table

Contents



Computer boting procedure

Boot procedure

- It will be a very high-level overview of the boot process. For more precise, see Operating systems course.
- User pressing the button, completing the electronic circuit, and providing the power to all.
- The CPU starts up but needs some instructions to work on (remember, the CPU always needs to do something). Since the main memory is empty at this stage, CPU defers to load instructions from the firmware chip on the motherboard and begins executing instructions
- The firmware code does a Power On Self Test (POST), initializes the remaining hardware, detects the connected peripherals (mouse, keyboard, pen drive etc.), and checks if all connected devices are healthy. You might remember it as a 'beep' that desktops used to make after POST is successful.
- Finally, the firmware code cycles through all storage devices and looks for a bootloader (usually located in the first sector of a disk). If the bootloader is found, then the firmware hands over control of the computer to it.

Boot procedure

- So now that the bootloader is loaded, its job is to load the rest of the operating system. GRUB is one such bootloader that is capable of loading unix-like operating systems and is also able to chain-load Windows OS. The bootloader is only available in the first sector of a disk, which is 512 bytes. Given the complexity of modern operating systems, some of these bootloaders tend to do multi-stage loading, where the main bootloader loads the second-stage-bootloader in an environment, which is not restricted to 512 bytes.
- The bootloader then loads the kernel into memory. Unix-like operating systems then run the init process (the master process, from which other processes are forked/executed) and finally initialize the run-levels.
- After all this, and after some other drivers are initialized, the Graphical User Interface (GUI) is loaded, and you are presented with the login screen.

BIOS VS UEFI

BIOS BOOTING

BIOS

BASIC INPUT/OUTPUT SYSTEM

MBR

MASTER BOOT RECORD

BOOT LOADER

KERNEL

OPERATING SYSTEM

UEFI BOOTING

UEFI

UNIFIED EXTENSIBLE FIRMWARE INTERFACE

EFI BOOT LOADER

KERNEL

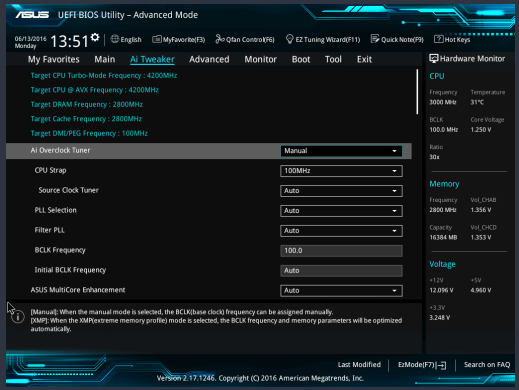
OPERATING SYSTEM

Bios

- BIOS is stored on an EPROM (Erasable Programmable Read-Only Memory), allowing the manufacturer to push out updates easily
- It provides many helper functions that allow reading boot sectors of attached storage and printing things on screen
- BIOS menu acces is platform-dependent, but usualy it's pressing 'Del' or 'F12' during the initial boot phase
- Intel is not supporting BIOS from 2020
- Uses only MBR partitions

UEFI

- UEFI does the same job as a BIOS
- Stores all data about initialization and startup in an .efi file, instead of storing it on the firmware
- This .efi file is stored on a special partition called ESP on the hard disk. This ESP partition also contains the bootloader
- UEFI supports bigger drive size, provides faster boot time, runs in 32-bit or 64-bit mode, can be used with GPT
- Has a Legacy mode - UEFI mode that behaves in a way as BIOS did



Bootloaders

Bootloaders overview

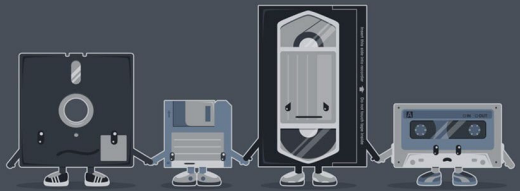
- Bootloader - a program responsible for booting the operating system
- Bootloader stored on a specific partition
- When PC is powered off, all data is stored on the non-volatile memory
- But for OS to work, it should be in the RAM
- Bootloader is responsible for loading the system to RAM
- There are too many platforms and too many ways how bootloaders could be used
- To simplify, here is an overview of: x86 architecture, modern Linux OS (kernel v5+)

GRUB

- GRUB - most wide-common used bootloader for our systems from the GNU project
- Can be used in both UEFI and Legacy modes
- If installed, during booting GRUB loads /boot/grub/grub.cfg file. Do not edit it!
- To change anything in that file, change it only in /etc/default/grub (remember, there are all setting files) and then use command gtub-mkconfig -o /boot/grub/grub.cfg
- But GRUB is too old and slow.
- More about it is on Arch wiki

systemd-boot

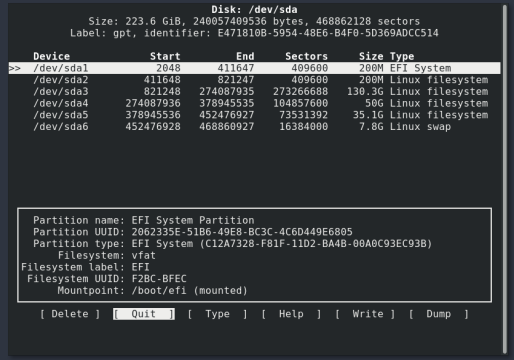
- Simple UEFI boot manager
- Executes configured EFI images
- Included with systemd
- Can only start EFI executables such as UEFI Shell, Linux EFISTAB , Windows, GRUB, Windows boot manager etc
- Can be installed only with ESP mounted
- Much faster than all other bootloaders (if your init system is systemd) which is true for both Manjaro and Arch (Ubuntu also)
- The reason - it starts fewer things in general and starts almost everything in parallel
- Simple in maintaining and setting up:
- bootctl install for installation
- /boot/loader/entries/ is a location for config files. If you want to add new entry, just add new .conf file here



PARTITION TABLES

Partitioning

- Linux see each device as a separate file a.e. /dev/sdX, /dev/nvme0nX
- Partitioning - adding logical devices remaining the same amount of phisical. It make possible to manage each partition separate
- Each such logical device called partition
- Partition table - area on a disk where the disk stores the information about every partition
- Partition editors can be used to add/delete, adit new partitions (a.e. fdisk, cfdisk, parted, gparted)



Sources

Sources

- UCU Linux Club resources
- Boot procedure
- Wiki Bootloader
- GRUB Arch Wiki
- Systemd-boot Arch wiki
- GRUB vs systemd-boot
- Partitioning Arch wiki