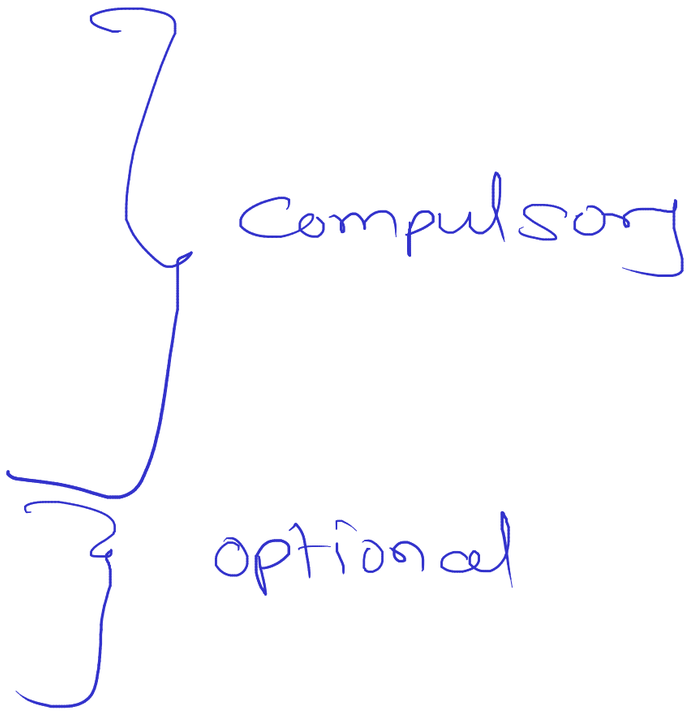
Operating System - Kernel

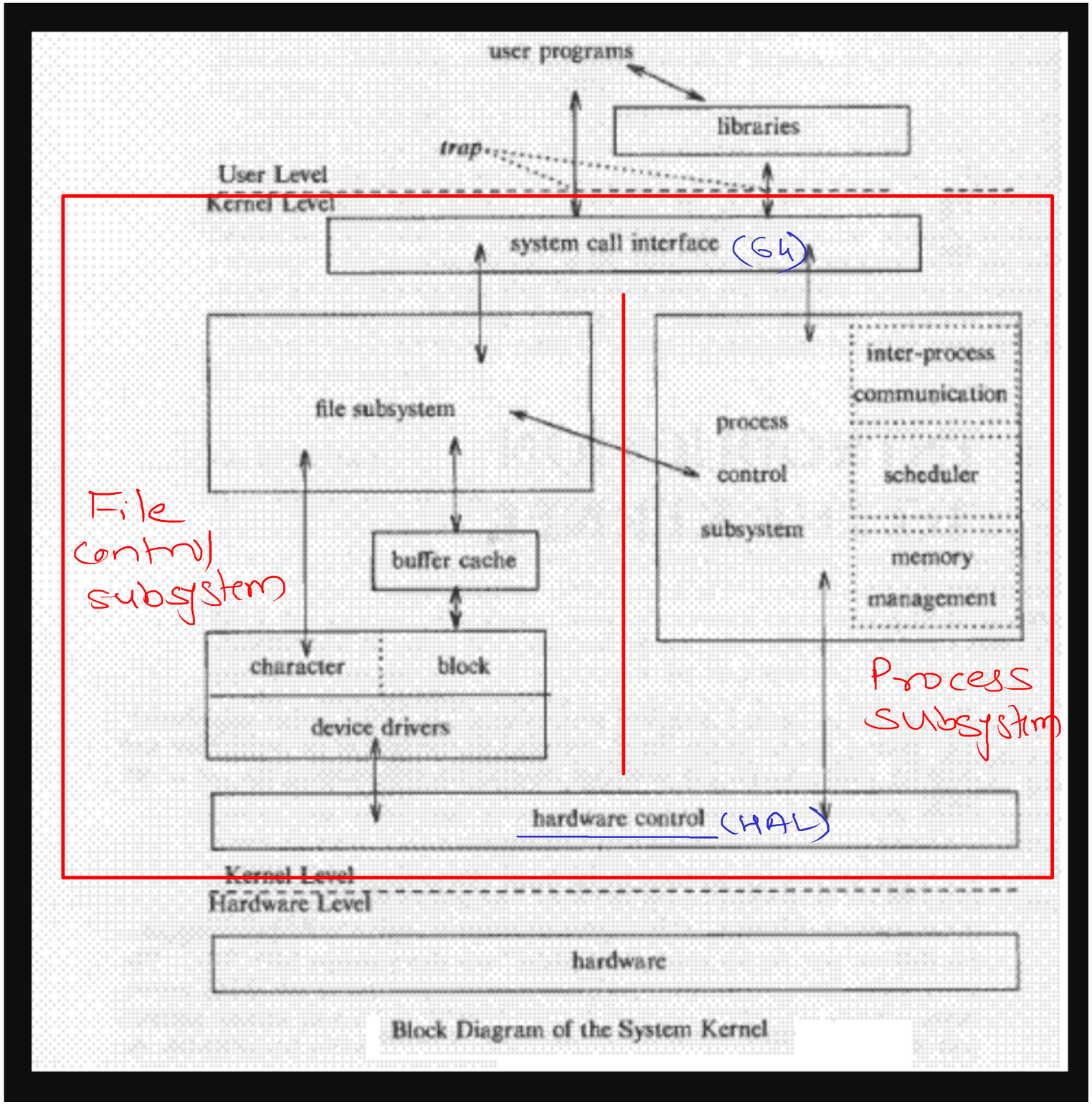
1) Process Management
2) CPU scheduling
3) Memory Management
4) File & IO Management
5) Hardware Abstraction
6) User Interfacing
7) Networking
8) Security & Protection



Types of Kernel

1) Monolithic Kernel
2) Micro Kernel
3) Modular Kernel
4) Hybrid Kernel
5) Nano Kernel

UNIX Architecture

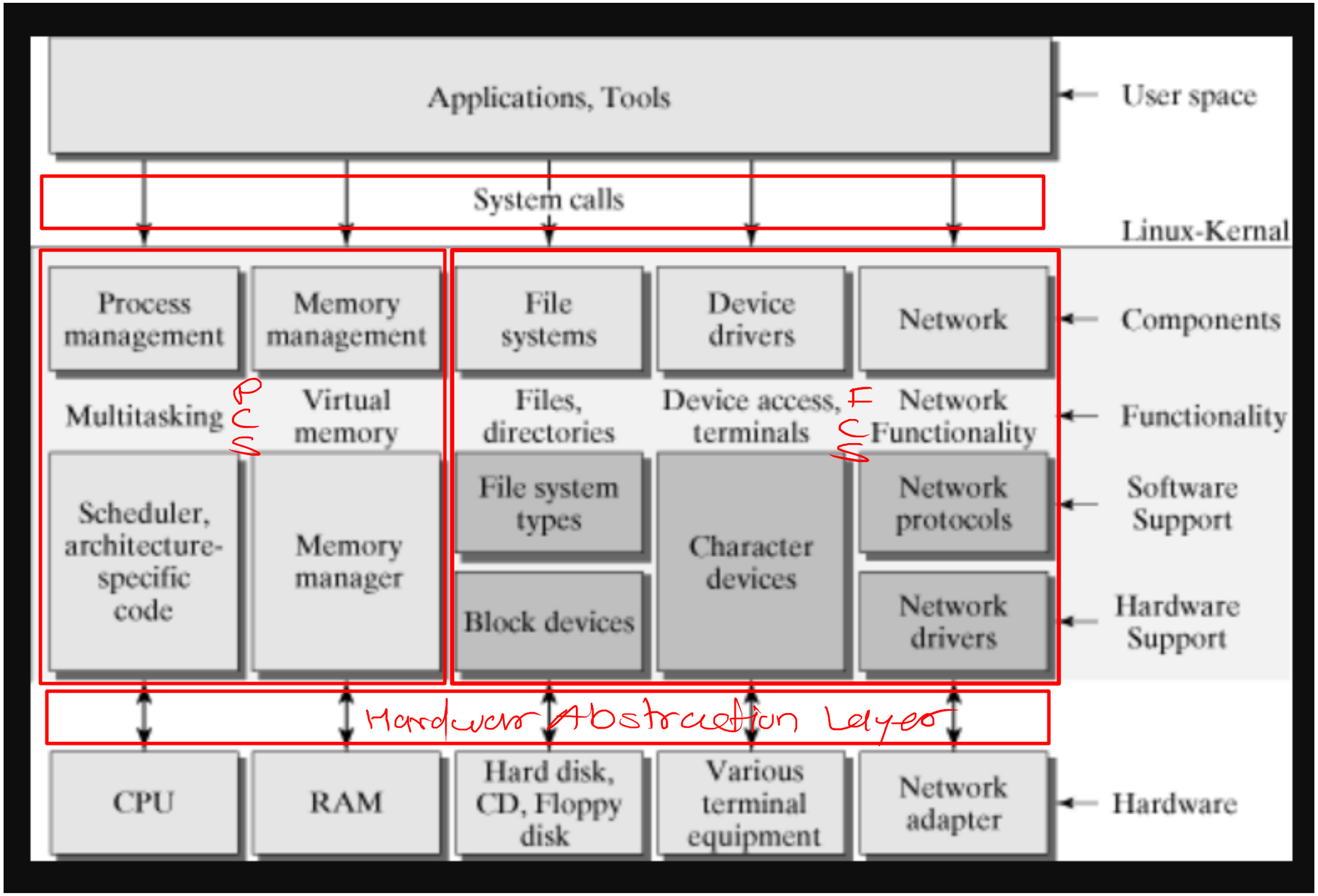


Block Diagram of the System Kernel

File Types

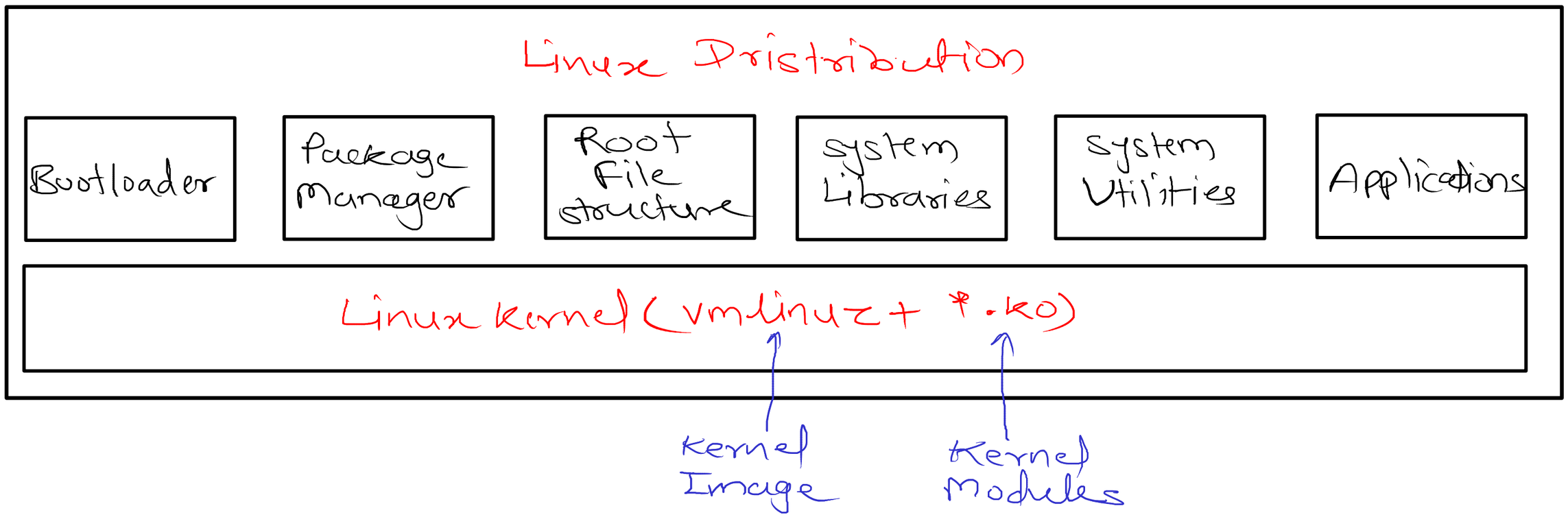
1) Regular (-)
2) Directory (d)
3) Link (l)
4) Pipe (p)
5) socket (s)
6) char special (c)
7) block special (b)

Linux Architecture



Linux Distribution

- Linux source code is available on www.kernel.org.
- Companies (like RedHat, Novell, ...) or individuals download source code, complie and integrate with other components like bootloader, user interface, package manager, root file system, libraries, system utilities & applications to develop Linux distributions.
- There are thousands of Linux distros available (www.distrowatch.com)
- Linux kernel compilation is compiling Linux source code only. It should be ensured that compiled kernel should work well with rest of the components.



Linux Kernel

Static component (monolithic)

1) Process Management
2) CPU scheduling
3) Memory Management
4) IO Subsystem (lower)
5) Hardware Actraction (HAL)
6) System calls (340+)

Kernel Image
(vmlinuz)
Location : /boot

make bzImage – compile
make install – to copy kernel into boot dir

Dynamic component (modular)

1) File system Managers
2) Device Drivers

Kernel modules
(Dynamically loadable modules)
Linux = .ko (Kernel Object)
Windows = .sys

Location : /lib/modules/kversion

make modules – to compile
make modules_install
↳ to copy all .ko files into

Linux Kernel Versions

linux-x.y.z.tar.xz

x — major revision
  ↳ hardware (arch) change
  ↳ may not backword compatible

y — minor revision
  ↳ subsystem change/addition
  ↳ backword compatible

z — revision
  ↳ bug fixes / patch

-generic — local version
  ↳ to identify the kernel

eg.

linux-5.15.116-DriversLab

5 — major revision
15 — minor revision
116 — revision
-DriversLab — local version

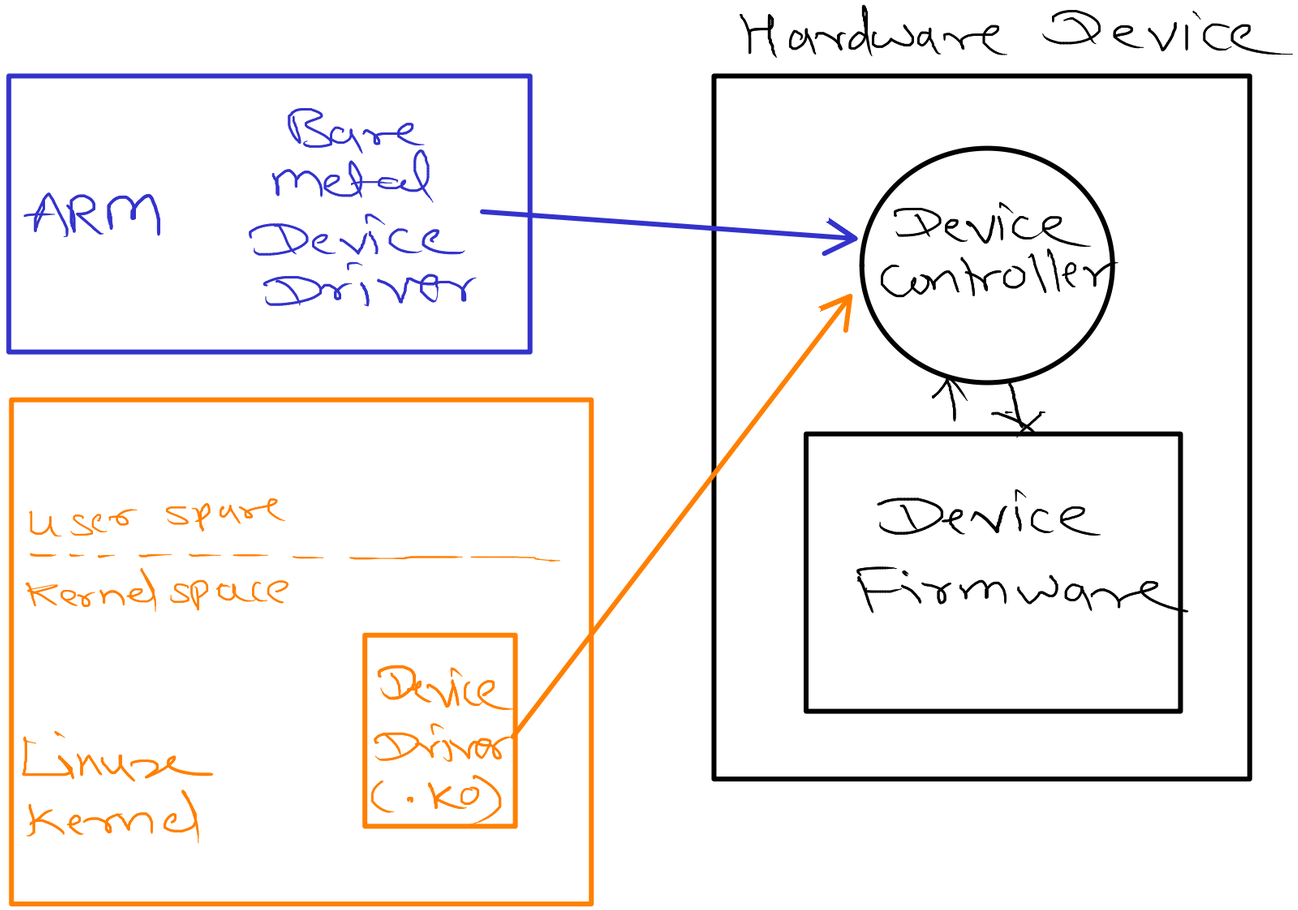
Linux command - tar to extract kernel archive

tar -xvf linux-5.15.116.tar.xz

-x -> extract
-v -> verbose
-f -> filename
-g / j / p (optional) -> g (gnu zip) .tar.gz
-> j (binary zip) .tar.bz2
-> p (extended zip) .tar.z

Linux device drivers

- Device driver is a kernel module that instructs device controllers to perform the operations and also handles interrupts generated from it.



Linux device drivers

* Character device drivers (raw device drivers)

- Char devices transfer data in byte by byte manner. So device drivers are implemented to read/writer data as stream of bytes. They support four major operations i.e. open(), close(), read() and write(). Example: Serial port, parallel port, keyboard, tty, etc.

/dev
↑
devfs
↑
created at runtime into RAM (not persistant)

* Block device drivers (cooked device drivers)

- Block devices transfer data as bunch of bytes i.e. block by block. Size of block is typically 512 Bytes. Support major operations open(), close(), read(), write() and lseek(). Example: All mass storage devices.

* Network device drivers

- Network drivers are responsible for packets transmit and receive, however network protocols are implemented up in network stack. Unlike character and block devices network device entry is not done under /dev.

Linux kernel compilation

- Linux kernel is monolithic.
- But it exhibits modular and micro-kernel nature as well.

    - Monolithic kernel image: vmlinuz (/boot)
    - Kernel modules: .ko (/lib/modules/<kernel-version>)

- Kernel source tree contains source code corresponding to kernel & modules.
    - arch, init, kernel, ipc, crypto, include, lib, mm, net, block, fs, drivers, sound, usr, scripts, ...

- Kernel release tree contains compiled kernel image & modules in root file system.
    - boot, lib, bin, sbin, home, usr, home, etc, ...

Kernel compilation steps

1. copy preconfigured .config file into source tree

2. make menuconfig
   - local version + some config
3. make bzImage
   - compile static component → ~src/arch/x86/boot/vmlinux → vmlinuz
4. make modules
   - compile dynamic component → *.ko
5. sudo make modules_install
   - copy *.ko to /lib/modules/kversion
6. sudo make install
   - copy vmlinuz to /boot

Kernel Configurations

1) make defconfig
   - default configuration file used

2) make config
   - questions are asked to user

3) make menuconfig – character based UI
   make gconfig – GTK UI
   make xconfig – QT

4) copy config file from boot directory
   cp /boot/config-<kversion> .config

[ ] – do not compile
[*] – compile module as static
[m] – compile module as dynamic

In makefile
obj-n := comp.o
obj-y := comp.o
obj-m := comp.o