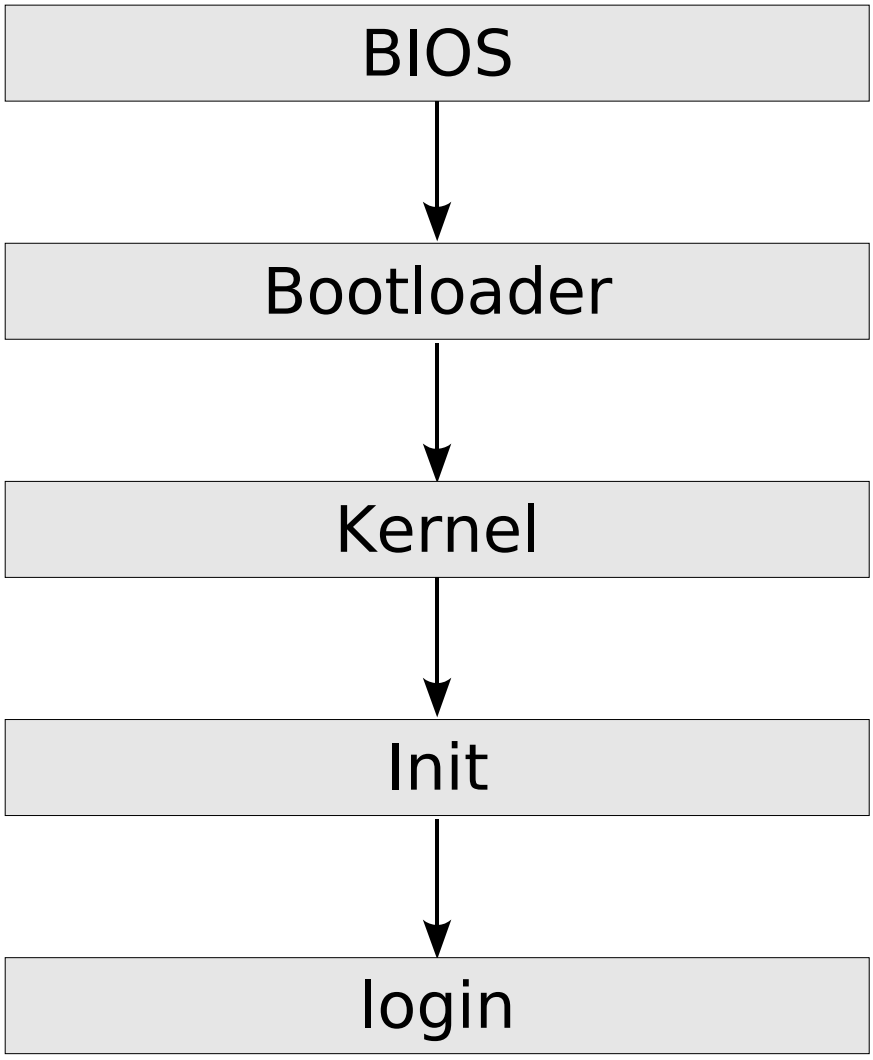
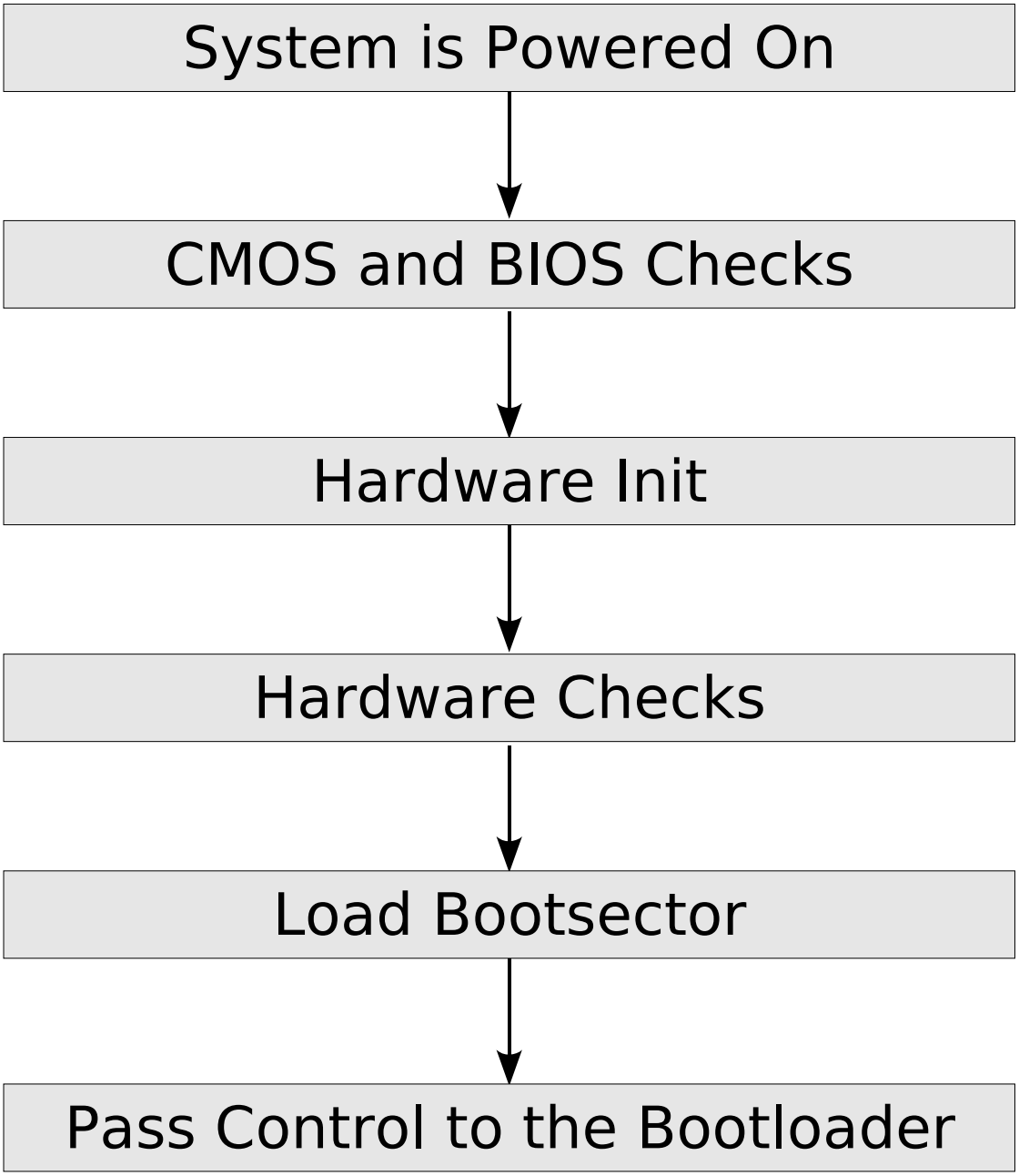
From Power to Command Line



BIOS (Basic Input/Output System)

This is the first thing that is run when the system is powered on. The BIOS combined with the CMOS (Complementary Metal Oxide Semiconductor) are responsible for Initializing hardware and running basic system checks on boot before handing the system over to the bootloader.



After the BIOS is done initializing the hardware then the system load the information that is in the bootsector into memory, and passes control of the system over to the bootsector code.

There are two commercial BIOS companies left Phoenix and AMI (American Megatrends Inc), since Award got baught by Phoenix. But there are some opensource projects in the works to make a more open and modifiable BIOS. Such as the LinuxBIOS Project, and the OpenBIOS Project.

The LinuxBIOS Project[1] involves booting the Linux Kernel as the BIOS. Modern commercial BIOS still have to be backwards compatible with the systems of old, and therefore have lots of old legacy code to be able to boot older operating system like DOS. Most modern kernels don't use any info from the BIOS and even do their own hardware initializing. Their final plan is to be able to boot linux from the BIOS chip and have linux exec another ELF format kernel. It could cut out the legacy bootloader method of loading the kernel altogether.

The OpenBIOS Project[2] is a implementation of the OpenFirmware[3] specification developed by the IEEE and use by Sun, IBM, Apple. The working on implementing a fully IEEE 1275-1994 complaint firmware that can be used on all common platforms. They're working closely with the LinuxBIOS Project.

Bootloader

The bootloader is the next stop in the system boot process. Its job is to load the kernel from somewhere on the system. Common bootloaders these days are Lilo[4] and GRUB[5] for linux, but there are many other bootloaders out there for all different architectures. Linux can be loaded from almost any bootloader. Some people even use the Windows 2000/XP bootloader to load the system.

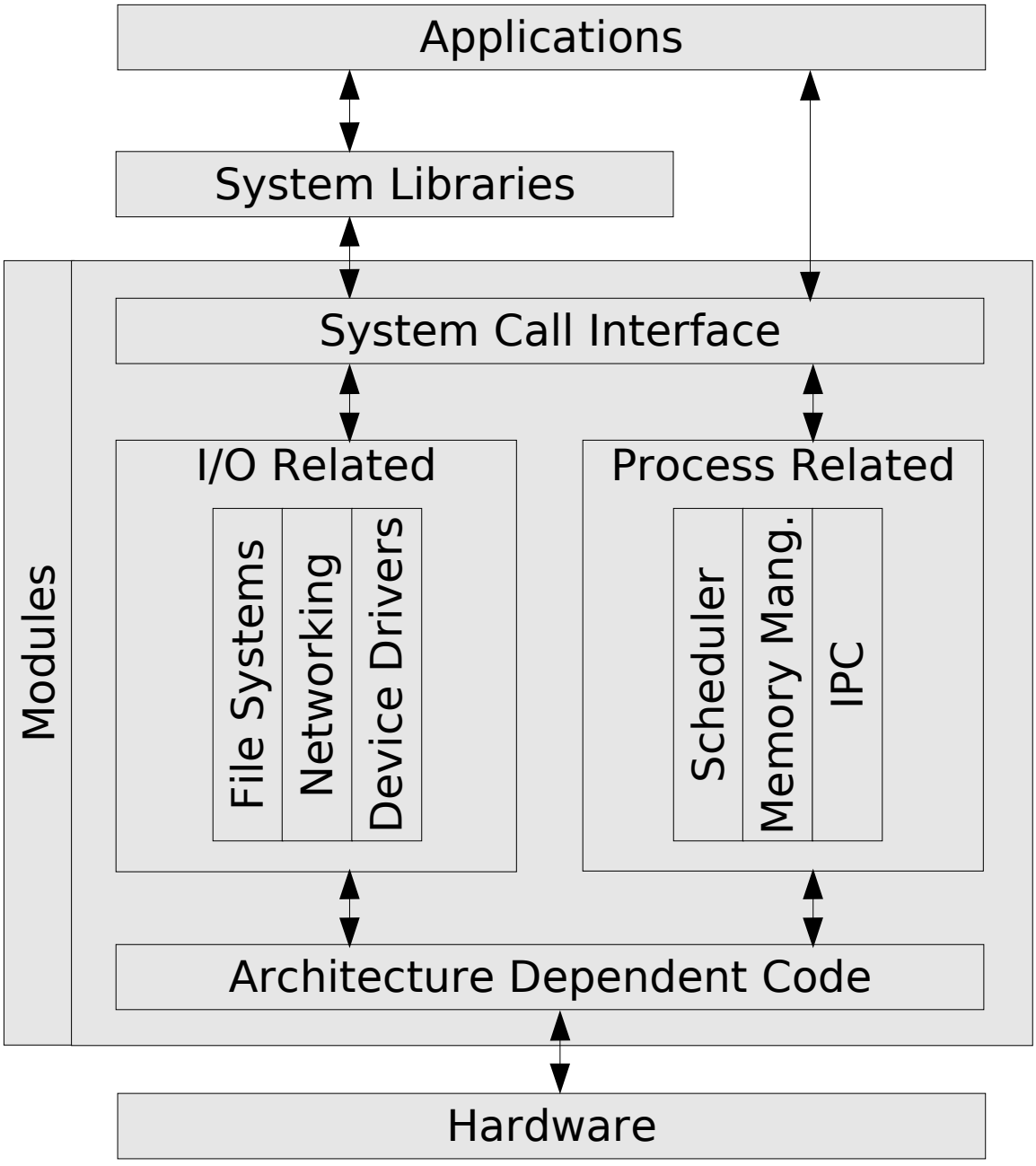
One of the earliest and still pretty popular bootloaders is Lilo. Lilo is a simple bootloader, with the main part of the application in residing in the boot sector. Lilo loads the kernel from the system by logging the exact location of the kernel on the hard disk. Doing this allows Lilo to operate without knowing anything about the file systems involved.

GRUB (GRand Unified Bootloader) is a multi-part bootloader. That means that there is only a small bootloader that sits in the bootsector and the rest is loaded off the system. There usually isn't enough room for the whole bootloader because the boot sector is only 512 bytes big.

In some area GRUB lacks some functionality and some flexibility. So the people on the GRUB development team are currently working on a new bootloader to supplement GRUB called PUPA[6]. It's still in the eairly stages but should support multiple partition formats like LVM, and many more file systems.

Kernel

The kernel is the core of the operating system. The kernels job is to manage the resources of the computer system and provide an interface for userland programs to work with resources and hardware. Its job on startup is to load drivers then run the init program commonly found at /sbin/init on the root partition.



Init

This is the area where the init scripts get loaded. There are many types of init programs providing varying services. Some are just run on script and load the terming login program, and others have many scripts with dependency checking.

The SystemV Init sequence is one of the most commonly used of all the init script system. It's used by RedHat, SuSE, Mandrake, Debian, Gentoo, and many others. The SystemV init sequence involves the inittab file which list what scripts to run on each 'runlevel'.

Runlevel	***Use***
0	Halt
1	Single User Mode
2	Multuser Mode (without networking)
3	Multuser Mode (with networking)
4	*User Customisable, not used*
5	Graphical
6	Reboot

The directory for the scripts vary from distribution to distribution. In a SystemV init system entering run level 3 the files in the /etc/rc3.d/ starting with a S are run with the start option. On a system exiting runlevel 3 and going to another run level the scripts beginning with a K are run with the stop option.

The file in the /etc/rc3.d directory are usally not the scripts themselves but symlinks to the real script that are in a init.d type directory. All these directory names are usally different on each distribution but the general concept is the same.

Distribution	***init.d***	***rc?.d***
RedHat	/etc/rc.d/init.d/	/etc/rc.d/rc?.d/
SuSE	/etc/init.d/	/etc/init.d/rc?.d/
Mandrake	/etc/rc.d/init.d/	/etc/rc.d/rc?.d/
Debian	/etc/init.d/	/etc/rc?.d/
Gentoo	/etc/init.d/	/etc/runlevels/{boot, default}

```
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Things to run in every runlevel.
ud::once:/sbin/update

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
# xdm is now a separate service
x:5:respawn:/etc/X11/prefdm -nodaemon
```

Basic layout of a inittab file

The BSD Init sequence is common in Slackware, and embedded systems. This method of boot script is pretty straight forward and the most simple method. The init can use a single script or have different scripts for each runlevel. Slackware runs /etc/rc.d/rc.S when the system boots and runs /etc/rc.d/rc.M when entering multiuser mode (ie. Runlevel 3).

Links

[1] LinuxBIOS Project: http://www.linuxbios.org/
[2] OpenBIOS Project: http://www.openbios.org/
[3] OpenFirmware: http://playground.sun.com/pub/p1275/

[4] Lilo: http://lilo.go.dyndns.org/
[5] GRUB: http://www.gnu.org/software/grub/grub.html
[6] PUPA: http://www.nongnu.org/pupa/