**DRIVERS BASICS**

WHAT IS DRIVER? Why driver?

* The software that handles or manages a hardware controller is known as a device **driver**. The **Linux** kernel device **drivers** are, essentially, a shared library of privileged, memory resident, low level hardware handling routines.
* In computing, a **device driver** is a computer **program** that operates or controls a particular type of **device** that is attached to a computer. ... **Drivers** are **hardware** dependent and operating-system-specific. They usually provide the interrupt handling required for any necessary asynchronous time-dependent **hardware** interface.
* A **driver** is software that allows your computer to communicate with hardware or devices.
* A device driver (often referred to as driver’) is a piece of software that controls a particular type of device which is connected to the computer system. It provides a software interface to the hardware device, and enables access to the operating system and other applications. There are various types of drivers present in GNU/Linux such as Character, Block, Network and USB drivers.

In this column, we will explore only character drivers.  
Character drivers are the most common drivers. They provide unbuffered, direct access to hardware devices. One can think of character drivers as a long sequence of bytes — same as regular files but can be accessed only in sequential order. Character drivers support at least the open(), close(), read() and write() operations. The text console, i.e., /dev/console, serial consoles /dev/stty\*, and audio/video drivers fall under this category.  
To make a device usable there must be a driver present for it. So let us understand how an application accesses data from a device with the help of a driver. We will discuss the following four major entities.

* User-space application: This can be any simple utility like echo, or any complex application.
* Device file: This is a special file that provides an interface for the driver. It is present in the file system as an ordinary file. The application can perform all supported operation on it, just like for an ordinary file. It can move, copy, delete, rename, read and write these device files.
* Device driver: This is the software interface for the device and resides in the kernel space.
* Device: This can be the actual device present at the hardware level, or a pseudo device.

Why clock used?

In synchronous communication used “synchronous” data bus, which means that it uses separate lines for data and a “clock” that keeps both sides in perfect sync. The clock is an oscillating signal that tells the receiver exactly when to sample the bits on the data line. This could be the rising (low to high) or falling (high to low) edge of the clock signal; the datasheet will specify which one to use. When the receiver detects that edge, it will immediately look at the data line to read the next bit (see the arrows in the below diagram). Because the clock is sent along with the data, specifying the speed isn’t important, although devices will have a top speed at which they can operate (We’ll discuss choosing the proper clock edge and speed in a bit).