What are Device drivers?

A device driver is a special kind of software program that controls a specific hardware device attached to a computer. Device drivers are essential for a computer to work properly.

Types of Device Drivers

- printer driver software.
- scanner driver software.
- Windows drivers.
- Linux drivers.

The Device drivers are special programs which control the execution of a particular hardware device. Whenever we purchase a new hardware for eg :- a printer from the manufacturer HP we will need to install the printer hardware with them installation disk. This will load the new driver into the operating system and enables us to use the hardware. Similarly each hardware example a monitor, scanner, Mouse, keyboard, modem etc. has its own driver. Devices from different manufacturers need their own drivers to function properly. Therefore it is of great importance.

The main purpose of device drivers is to provide abstraction by acting as a translator between a hardware device and the applications or operating systems that use it. Programmers can write higher-level application code independently of whatever specific hardware the end-user is using.

Device Drivers are the software through which the kernel of a computer communicates with different hardware without having to go into the details of how the hardware works. It is software that controls a hardware part attached to a computer and allows it to use the hardware by providing a suitable interface. This means that the operating system need not go into the details about how the hardware part works. It also provides a common interface so that the operating system or the Kernel can communicate with the hardware. Thus, the purpose of device drivers is to allow smooth functioning of the hardware for which it is created and to allow it to be used with different operating systems.

Difference between RTOS and GPOS

RTOS	GPOS
RTOS has unfair scheduling i.e scheduling is based on priority.	GPOS has fair scheduling i.e it can be adjusted dynamically for optimized throughput.
Kernel is pre-emptive either completely or up to maximum degree.	Kernel is non-preemptive or has long non-preemptive code sections.
Priority inversion is a major issue.	Priority inversion usually remains unnoticed.
It has a predictable behaviour.	There is no predictability.
It works under worst case assumptions.	It optimizes for the average case.
It does not have a large memory.	It has a large memory.
It is used for dedicated electronic application	It is used for general universal application

RTOS	GPOS
There is a task deadline in RTOS	There is no task deadline in GPOS
The Time response of RTOS is deterministic	The time response of OS is not deterministic
Depending upon application we can customize the RTOS	Depending upon application, we cannot customize the GPOS
It optimizes the memory resources	It does not optimizes the memory resources.
It is normally stored in semiconductor memory like EEPROM, Flash EEPROM	It is normally stored in Hard disk
The applications are compiled and linked with the RTOS.	The applications are compiled and linked separately from the operating system.

Difference between general purpose system vs embedded systems.

General Purpose Computer	Embedded Systems
It is designed using a microprocessor as the main processing unit.	It is mostly designed using a microcontroller as the main processing unit.
It contains a large memory semiconductor memories like cache and RAM. it also contains secondary storage like hard disks etc.	It uses semiconductor memories, but does not require secondary memories like hard disk, CD. It sometime has special memory called flash memory.
It is designed such that it can cater to multiple tasks as per requirement.	It is designed such that it can cater to a particular predefined task.
It is mostly costlier compared to the embedded systems	It is cheaper compared to a computer.
It requires huge number of peripheral devices and their controllers	It is cheaper as it requires less no of peripheral devices and their controllers are microcontroller chip itself.
The Operating system and other software for the general purpose computers, are normally	The operating system(mostly RTOS i.e Real Time Operating System) and

General Purpose Computer	Embedded Systems
complicated and occupy more memory space	other software occupy less memory space.

How can hardware understand the code we write in embedded systems (.c file to .exe file)

The code the user writes is translated into a set of 1’s and 0’s by a compiler. All the compiler understands is “high” and “low” voltages, or 1’s and 0’s. Each instruction generated by the compiler is executed in a cycle. The hardware accesses the memory to retrieve an instruction. Hardware understands code through the use of a specific type of software called an operating system. The operating system acts as an intermediary between the hardware and the code, translating the code into instructions that the hardware can understand and execute. These instructions are typically in the form of machine code, which is a series of binary digits (ones and zeroes) that the hardware can interpret. The operating system also manages the resources of the hardware, such as memory and processing power, to ensure that the code runs efficiently. The code the user writes is translated into a set of 1’s and 0’s by a compiler. All the compiler understands is “high” and “low” voltages, or 1’s and 0’s. Each instruction generated by the compiler is executed in a cycle.