Difference b/w General Purpose s/m & an embedded s/m:

General Purpose s/m:-

It is a computer s/m that can be programmed to perform a large no. of tasks.

The ability to run many different pieces of software allows a general purpose s/m to be quite versatile in terms of the types of tasks it can perform.

Typically, a general purpose s/m has a wide range of i/ps & o/ps that can be connected to it.

Ex: including USB ports on a laptop allows other devices to change the capabilities & features available to the laptop

Embedded s/ms:-

It is an computer s/m that carry out a small number of tasks.

When designing an embedded s/m, manufacturers will focus on the dedicated func that the s/m need to perform. They will optimise the s/m until it performs each of these tasks very efficiently.

Modern embedded s/ms contain a microcontroller, which ~~contain~~ a mic consists of a CPU to process data, as well as a fixed amount of RAM & ROM. Earlier embedded s/ms were based on microprocessors that contained only the CPU.

Advantages of ES:-

* Highly efficient at performing tasks
* Extremely reliable
* Easy to design
* Cheap to produce
* Compact in size
* Low power consumption

2) What are device drivers?

Device drivers are the software libraries that initialize the hardware & manage access to the hardware by higher layers of software

may

A device driver is a particular form of software application that allows one Hardware device (PC) to interact with another hardware device (Printer).

Device drivers faciliate communication b/w an Operating s/m & a Peripheral hardware device.

Today, most Operating Sims include a library of Plug-n-play drivers that allows Peripheral hardware to connect automatically with an OS.
There are various types of device drivers for I/O devices such as keyboards, CD/DVD, controllers, printers. When a driver is included in an OS, it may be refered to as a kernal-mode device driver.

, scanner driver, Windows, Linux.

3) How can Hardware understand the codes that we write in an embedded sim?

All the code the user writes is translated into a set of 1's & 0's by a Compiler. All the Computer understands is high & low voltages or 1's & 0's. Each instruction generated by the compiler is executed in a cycle. First the hardware access the memory to retrieve an instruction

⇒ CPU is at the heart of the computer. It only understands something called machine code → 1's & 0's.

⇒ binary code is the only language that computer hardware can understance. Each CPU has it's own specific machine language

4) What is RTOS & General purpose operating sim (GPOS)?

RTOS → It is an OS that guarantees real time applications a certain capability within a specified deadline.

⇒ RTOS are designed for Critical sims & for devices like microcontrollers. RTOS Processing time requirements are measured in milliseconds.

Ex:- Air line traffic Control, Commaral control sims, Airlines reservation sim, Heart pacemaker, Robot.

Types:- Hard RTOS
Soft RTOS
firm RTOS

GPOS:- It can run on various platforms & devices, & support a wide range of applications & features.

It is an essential component of any mobile device, server/computer & is responsible for running all the applications in an installation

ex:- Linux, Windows, IOS

Differences:-

* A higher priority thread can't Preempt a kernal call in GPOS a low priority job in an RTOS would be preempted by a higher priority one, even executing a kernal call.
* RTOS used dedicated electronic application,
  GPOS used general universal applications
* RTOS → single user environment → deterministic.
  GPOS → multi user enviroment → not deterministic
* RTOS → optimizes memory resources
  GPOS → not
* RTOS → task deadline
  GPOS → no task deadline
* RTOS mainly used in ES (small), GPOS mainly used in PC, servers, tablets, mobile (large memory)

↓
C language