1. What are the main features of Real-Time Operating Systems and how do they differ from regular OS’s?

**Answer**:

The main feature of Real-Time Operating System (RTOS) is have guaranteed response time to external events. The use of RTOS is always associated with the hardware, the object and the events taking place on the object. It is like a hardware-software complex and usually includes sensors, input-output modules. The RTOS is focused on handling external events. It is this main difference from the regular OS

1. When is it preferable to use STM32 over Arduino, ESP32, PIC or other comparable embedded system?

**Answer**:

Microcontrollers STM32 have a large series of models for any task. You can choose a chip for your tasks. Also STM32 are pin-to-pin compatible meaning you can easily migrate to more powerful or with extended peripherals with minimal code modifications. So I choose STM32 when I need project flexibility, development speed and the availability of peripherals that Arduino does not have (for example DFSDM module).

1. Let's assume that a few of the peripherals you use have the same hardcoded I2C address. What solution would you use to solve this problem?

**Answer**:

Decision will depend on the situation. I can suggest:

* + set a switch that will connect the device to the line at the time of data exchange;
  + set retranslator with other address between line and device.

1. Let's assume that there is a hardware module that is attached to the motherboard with a mezzanine connector. Due to the small form factor, the number of pins on the mezzanine connector is much less than the number of pins you need. Assuming that the communication speed is not important, how would you solve this problem?

**Answer**:

If there is a task of transmitting logical signals I prefer to use multiplexer / demultiplexer. Also for communication between motherboard and hardware module can be used interface such as CAN, RS485, I2C.

1. Let's say you want about 30 hardware modules to communicate with each other. There is one STM32 microcontroller on each module, and the modules can be removed and installed instantly. Which communication standard would you use for these modules to communicate effectively with each other? Why?

**Answer**:

I prefer use CAN interface. It is a industrial interface, with speed up to 1 Mbps and has advantages:

- Allows you to control that the received data is correct (have a CRC)

- You can set up a filter for certain IDs, and don’t react on others ID

- Has an arbitration system.

1. 3 devices with identical embedded software and hardware are required to communicate over I2C. What kind of solution would you develop so that devices can be dynamically addressed and recognize each other?

**Answer**:

I would use two I/O ports to determine the address. Depending on the binary combination of signals (00, 01,10, 11), the device could understand its purpose.