**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?**

**Solution 1**

This is simpliest solution for me using other I2C bus. If we have enough hardware that supply another I2C bus I think this is most preferable solution.

**Solution 2**

Control sda and scl line of devices by gpio pins to distinguish same addressed devices. With this way we must write seperate I2C transmit or receive function for each same addressed I2C devices. For example we have 2 same addressed devices(Device-6 Device-13). When I want to communicate device-6 before I send the address I must break the device-13 from I2C Data bus with transistor etc. This solution has much disadvantage on hardware and software. It causes software complexity and requires extra gpio pin.

**Solution 3**

For this problem, we can solve by using extra hardware like I2C multiplexer or I2C address translator. I think these are most reliable but costly way.

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

If we work on professional project, we must firstly eliminate Ardunio.

We can prefer ESP32, if we work on ethernet based project and it is simple.

If we work on professional, price efficient, multi-peripheral used project, we must use STM32. STM32 is preferable for professional work over Ardunio, ESP32, PIC. ST give us a wide range of mcu that have wide kind of peripheral and tools. We can select suitable mcu for our project. STM32 is very populer in industry so we can reach much resource(Libraries, error fixing, how to do, etc.) about it.

**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?**

If there is a multiplexer for mezzanine connector, I would use it. If we use same communication protocol like I2C, UART, SPI, digital pins. I would find the communication and digital pins and use multiplexer for this pins to accommodate my required pins.

**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?**

First of all, I didn't know solution to this problem because I know that if a module is connected to the bus without power, the bus will not work. I researched this question and found that SMBus is suitable for this type of communication because of SMBus’s how swappable feature.

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

RTOS is a OS that is designed to run time critical and high reliability required applications. It seperate a certain time for each processes(time scheduling).

İmportant difference between regular os and rtos;

Regular OS are not based on task scheduling and Regular OS can run more task which is not time critical. In rtos priority of process can effect of processing, but Regular OS are based on tailing in terms of priority.