Assignment 10 - Real Time Operating Systems Results for Harish Marepalli (He/Him)

Score for this attempt: **20** out of 20
Submitted May 7 at 11:41pm
This attempt took 3 minutes.

Question 1

6 / 6 pts

List the three classifications of Real Time Systems, and briefly describe each

Your Answer:

Real-time systems can be classified into three categories based on their response time requirements and consequences of missing deadlines. They are as below.

1. Hard Real-Time Systems

2. Soft Real-Time Systems

3. Firm Real-Time Systems

Description:

1. Hard Real-Time Systems: A hard real-time system causes an entire system to fail if it misses its deadline or time constraint. In hard real-time systems, missing a deadline can result in catastrophic consequences such as injury or loss of life. These systems must provide a guaranteed response time for critical tasks, and the system is considered to have failed if the response time exceeds the specified deadline.

Example: 1. A flight control system with an unacceptable latency may cause an aircraft to crash (Control systems for aircraft and nuclear power plants).

2. Soft Real-Time Systems: With these apps, results degrade after their deadline, whether the deadline is met or not. In soft real-time systems, missing a deadline may not result in catastrophic consequences, but it can lead to degraded system performance or reduced system efficiency.

These systems have response time requirements that are not as strict as those in hard real-time systems.

Example: 1. A video game is an example of a soft real-time system. Video games rely on user input and have limited time to process; degradation is sometimes expected for this reason (Multimedia applications and video games).

3. Firm Real-Time Systems: With this type of app, a missed deadline is tolerable but causes significant degradation in quality. Firm real-time systems have response time requirements that fall somewhere between hard and soft real-time systems. These systems have deadlines that are important to meet, but missing a deadline does not lead to catastrophic consequences. However, there may be a cost associated with missing deadlines, such as a reduction in revenue or customer satisfaction.

Example: 1. In video conferencing, latency may degrade the quality of a call, but the system is still usable.

2. Stock trading and online transactions.

Question 2

7 / 7 pts

Match the characteristic with the type of system

Correct!	**Designed to handle a mix of tasks**	GPOS
Correct!	**Guaranteed response times**	RTOS
Correct!	**Dynamic memory**	GPOS
Correct!	**Jitter is tens of microseconds**	RTOS

Correct!

	Scheduling optimized for throughput	GPOS
Correct!	**POSIX 1003.1b**	RTOS
Correct!	**Virtual memory**	GPOS

Question 3

5 / 5 pts

List the mitigation strategy for each type of RTOS failure

Correct!	**Load Exceeded**	Scalable resources
Correct!	**Hardware Failure**	Redundant hardware
Correct!	**Software error**	Recovery algorithm
Correct!	**Deadlocks**	Abort & restart processe
Correct!	**Non-Recoverable Failure**	Restart system

Question 4

2 / 2 pts

Code in an RTOS may be optimized for both size and performance. Explain why.

Your Answer:

Code in an RTOS may be optimized for both size and performance because these two factors are often interdependent and affect each other in different ways. In general, the smaller the code size, the faster the code can execute, but smaller code size may also lead to reduced functionality and flexibility.

Optimizing for size: It means reducing the amount of memory required to store the code and data of an application or system. This can be important for systems with limited memory resources, such as embedded systems or systems with strict memory constraints. By reducing the size of the code, more memory is available for data storage, and the system can run more efficiently. In addition, smaller code size can also improve the speed of the system, as there are fewer instructions to execute.

Optimizing for performance: It means improving the speed and responsiveness of the system. This can involve techniques such as code profiling, optimization of frequently executed code sections, and minimizing the number of context switches or interrupts. By optimizing for performance, the system can respond more quickly to real-time events and handle more tasks within a given time frame.

In an RTOS, optimizing for both size and performance is important because the system needs to respond quickly to real-time events while also operating within memory constraints. Therefore, RTOS developers need to balance the trade-offs between code size and performance to ensure that the system can operate effectively while meeting its real-time requirements.

Quiz Score: **20** out of 20